A non-intrusive proper generalized decomposition scheme with application in biomechanics

X. Zou[1,2], M. Conti[1], P. Díez[2,*,†] and F. Auricchio[1]

[1]*Dipartimento di Ingegneria Civile ed Architettura (DICAr), Università degli Studi di Pavia, Via Ferrata 3, 27100 Pavia, Italy*
[2]*Laboratori de Càlcul Numric (LaCàN). E.T.S. de Ingenieros de Caminos, Canales y Puertos, Universitat Politècnica de Catalunya (UPC), Jordi Girona 1-3 E-08034 Barcelona, Spain*

SUMMARY

Proper generalized decomposition (PGD) is often used for multi-query and fast-response simulations. It is a powerful tool alleviating the curse of dimensionality affecting multi-parametric partial differential equations. Most implementations of PGD are intrusive extensions based on in-house developed finite element (FE) solvers. In this work, we propose a non-intrusive PGD scheme using off-the-shelf FE codes (such as certified commercial software) as an external solver. The scheme is implemented and monitored by in-house flow-control codes. A typical implementation is provided with downloadable codes. Moreover, a novel parametric separation strategy for the PGD resolution is presented. The parametric space is split into two- or three-dimensional subspaces, to allow PGD technique solving problems with constrained parametric spaces, achieving higher convergence ratio. Numerical examples are provided. In particular, a practical example in biomechanics is included, with potential application to patient-specific simulation.

Received ...

KEY WORDS: Proper Generalized Decomposition, non-intrusive, model order reduction.

1. INTRODUCTION

Modern simulation-based engineering sciences require solving parametric PDEs, with the parameters ranging in some admissible intervals. Taking these parameters as extra-coordinates (independent variables) increases the dimensionality of the problem. For this reason, the exploration of the parametric space blows up the computational cost exponentially, leading to the so-called curse of dimensionality.

Proper generalized decomposition (PGD) [1, 2, 3, 4] is a recently developed technique for model order reduction, being particularly powerful on tackling the curse of dimensionality. Originally, it was devised to solve kinetic theory models involving the Fokker-Planck equation [5, 6], for which standard discretization techniques fail due to the huge computational requirements. The PGD is based on the idea of separation of variables. The parametric solution is assumed to have a separated representation, in order to reduce the multi-dimensional problem to a sum of tensor product of functions defined in lower dimensional subspaces. In the context of model order reduction, the quality of *a posteriori* methods such as Reduced Basis (RB) [7, 8, 9] and Proper Orthogonal Decomposition (POD) [10] can only be evaluated after the establishment of the reduced bases which

*Correspondence to: P. Díez, Laboratori de Càlcul Numric (LaCàN). E.T.S. de Ingenieros de Caminos, Canales y Puertos, Universitat Politècnica de Catalunya (UPC), Jordi Girona 1-3 E-08034 Barcelona, Spain
†E-mail: pedro.diez@upc.edu

we prefer to call modes. Unlike the aforementioned methods, PGD is an *a priori* reduction method [11, 12], that is to say, PGD is able to create the modes and measure the approximation error on-the-fly. In a practical implementation, PGD generates a computational vademecum [13, 14] in the offline phase, which can be frequently used in the subsequent online phase for different types of problems that require multi-queries and/or extremely fast responses. Readers may refer to [15, 16] for recent reviews on the PGD framework. In this paper, we will focus on the offline phase. The original problem is solved with an alternated directions scheme, generating a sequence of sectorized problems. Well-known numerical methods, such as finite element (FE), can be applied to solve the separated problems for the offline phase.

To the authors' knowledge, most of the practical PGD frameworks currently available in the literature are based on intrusive implementations relying on academical FE source codes, commonly requiring cumbersome coding work. Typical academical PGD codes can be accessed in [17]. The PGD is not usually implemented in commercial FE software, which is commonly used in industries. Motivated by accelerating the PGD coding procedure to provide a bridge between this academical achievement and its industrial applications, we propose a non-intrusive scheme that takes advantage of off-the-shelf FE solver as a black box, and solves the sectorized problems outside of in-house developed PGD codes. To make the implementation simple but practical, we have chosen Matlab (Mathworks, USA) as the in-house developed code, and Abaqus (Dassault Systèmes, France) as the external solver. To transfer data between the two codes, plain text files are used as a simple interface. This idea is particularly interesting for promoting the application of PGD in industries which mainly rely on commercial FE software that are usually not open source codes. A recent work by Courard *et al* [18] suggests that the non-intrusive PGD scheme is workable for shape optimization problems with geometrical parameters as extra-coordinates.

As a secondary contribution in this paper, we propose a novel strategy on the separation for the parametric problems. Instead of separating the parametric space as a tensor product of 1D subspaces, we separate it into (tensor products of) higher dimensional (2D or 3D) parametric subspaces which collect the highly correlated parameters. With this strategy of keeping some correlated parameters unseparated, we are able to solve problems involving parametric spaces that have low separability due to constraints from physics, geometry or other aspects.

It is especially interesting to apply PGD for patient-specific simulation in biomechanics. For instance, recent works [19, 20, 21, 22] studied the implementation of PGD in real-time simulation for haptic surgeries. In this work, we present several numerical examples based on linear elasticity model for demonstration of the non-intrusive PGD scheme as well as the new approach of separating the parametric space. A practical application in bone mechanics is taken into account. This application involves medical image-based numerical modelling, parameter identification for biological tissues from experimental data, and can be used in patient-specific real-time simulation for clinical decision making.

The remainder of the paper is structured as follows. Section 2 states the linear elasticity problem and introduces PGD application in the intrusive implementation. Section 3 details the non-intrusive PGD scheme, taking into account material properties or loading location as extra-coordinates. In Section 4, we propose the further application of the non-intrusive PGD scheme on constrained parametric space with the unseparated approach. Three numerical examples characterized by different levels of complexity are finally presented in Sections 5 and 6.

2. PROBLEM STATEMENT

2.1. PGD for linear elasticity problem with multiple parameters

In this section we review briefly the governing equations in linear elasticity for later extensions in PGD algorithms. For the application in biomechanics, the quantity we are interested in is the strain, because we have a set of strain values measured from experiments which will be detailed later.

Under the infinitesimal deformation assumptions, the relationship between strain tensor $\boldsymbol{\varepsilon}$ and displacement $\mathbf{u}(\mathbf{x})$ is

$$\boldsymbol{\varepsilon} = \nabla^{\mathrm{s}}\mathbf{u} = \frac{1}{2}\left[\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right]. \tag{1}$$

The stress tensor $\boldsymbol{\sigma}$ and the strain tensor $\boldsymbol{\varepsilon}$ are related with elasticity tensor $\mathbb{C}$ by Hooke's law

$$\boldsymbol{\sigma} = \mathbb{C} : \boldsymbol{\varepsilon}. \tag{2}$$

Consider an elastic domain $\Omega \subset \mathbb{R}^d, (d = 1, 2, 3)$, the function of interest is displacement $\mathbf{u}$, computed through a following boundary value problem (BVP). The strong form of the BVP is stated as follows: find $\mathbf{u}$ satisfying the equilibrium equation and the boundary conditions

$$\begin{cases} \nabla \cdot \boldsymbol{\sigma} + \mathbf{b} = 0, & \text{in } \Omega, \\ \mathbf{u} = \mathbf{u}_D, & \text{on } \Gamma_D, \\ \boldsymbol{\sigma} \cdot \mathbf{n} = \mathbf{t}_N, & \text{on } \Gamma_N, \end{cases} \tag{3}$$

where $\mathbf{b}$ represents the body force. The Dirichlet boundary displacement $\mathbf{u}_D$ and Neumann boundary traction $\mathbf{t}_N$ represent the boundary conditions.

Defining the trial function space $V := \{\mathbf{u} \in H^1(\Omega) : \mathbf{u} = \mathbf{u}_D \text{ on } \Gamma_D\}$ and the test function space $V_0 := \{\mathbf{u} \in H^1(\Omega) : \mathbf{u} = 0 \text{ on } \Gamma_D\}$, the standard weak form reads: find $\mathbf{u} \in V$ such that

$$a(\mathbf{u}, \mathbf{v}) = l(\mathbf{v}), \quad \forall \mathbf{v} \in V_0, \tag{4}$$

with the bilinear and linear forms $a(\cdot, \cdot)$ and $l(\cdot)$ given by

$$a(\mathbf{u}, \mathbf{v}) := \int_\Omega \nabla^{\mathrm{s}}\mathbf{u} : \mathbb{C} : \nabla^{\mathrm{s}}\mathbf{v}\, d\Omega \quad \text{and} \quad l(\mathbf{v}) := \int_\Omega \mathbf{b} \cdot \mathbf{v}\, d\Omega + \int_{\Gamma_N} \mathbf{t}_N \cdot \mathbf{v}\, d\Gamma. \tag{5}$$

In many practical problems, model parameters such as material properties, loading locations, are difficult to obtain exactly, while only their ranges are known from previously performed tests or from the literature. Consequently, it would be greatly helpful if we are able to obtain a parameterized solution by taking advantage of PGD. Following the standard PGD procedure for parameterized problems [23, 15, 13], we construct the extended weak form by assuming the parameters as extra-coordinates. The displacement $\mathbf{u}(\mathbf{x})$ is then generalized to $\mathbf{u}(\mathbf{x}, \boldsymbol{\mu})$, where $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_m) \in \Omega_\mu$ is the vector of m independent parameters. Let I_{μ_j} be the range of parameter μ_j, assume the parametric space is Cartesian, namely $\Omega_\mu = I_{\mu_1} \times \ldots \times I_{\mu_m}$, we extend the solution space to $\mathbf{u}(\mathbf{x}, \boldsymbol{\mu}) \in V \bigotimes_{j=1}^m L^2(I_{\mu_j})$.

The extended weak form of problem then reads: find the displacement $\mathbf{u} \in V \bigotimes_{j=1}^m L^2(I_{\mu_j})$, such that for all $\mathbf{v} \in V_0 \bigotimes_{j=1}^m L^2(I_{\mu_j})$:

$$\mathcal{A}(\mathbf{u}, \mathbf{v}) = \mathcal{L}(\mathbf{v}), \tag{6}$$

where the bilinear and linear forms are extended from (5):

$$\mathcal{A}(\mathbf{u}, \mathbf{v}) := \int_{I_{\mu_1}} \cdots \int_{I_{\mu m}} a(\mathbf{u}, \mathbf{v})\, d\mu_1 \cdots d\mu_m, \qquad \mathcal{L}(\mathbf{v}) := \int_{I_{\mu_1}} \cdots \int_{I_{\mu m}} l(\mathbf{v})\, d\mu_1 \cdots d\mu_m. \tag{7}$$

2.2. *PGD separated representation*

The general procedure of solving parametrization problems with PGD is discussed intensively in the literature [23, 15, 13, 24]. Here we briefly introduce the PGD separated representation to adapt our purpose.

One of the major important issues for PGD is the separability of the extended solution. The separated representation of model parameters, boundary conditions and/or the source terms is always required for an efficient numerical computation. In computation, the extended weak form

(7) is supposed to be separable, so that it can be factorized as:

$$\begin{aligned}\mathcal{A}(\mathbf{u},\mathbf{v}) &:= \int_{I_{\mu_1}}\cdots\int_{I_{\mu_m}} a(\mathbf{u},\mathbf{v})\,d\mu_1\cdots d\mu_m = \left(\int_{\Omega}\ldots d\Omega\right)\left(\int_{I_{\mu_1}}\ldots d\mu_1\right)\cdots\left(\int_{I_{\mu_m}}\ldots d\mu_m\right),\\ \mathcal{L}(\mathbf{v}) &:= \int_{I_{\mu_1}}\cdots\int_{I_{\mu_m}} l(\mathbf{v})\,d\mu_1\cdots d\mu_m = \left(\int_{\Omega}\ldots d\Omega\right)\left(\int_{I_{\mu_1}}\ldots d\mu_1\right)\cdots\left(\int_{I_{\mu_m}}\ldots d\mu_m\right).\end{aligned} \tag{8}$$

To guarantee the factorization (8), the extended solution for the parameterized problem is assumed to be approximated by a superposition of n modes

$$\mathbf{u}(\mathbf{x},\boldsymbol{\mu}) = \mathbf{u}^n_{\text{PGD}}(\mathbf{x},\boldsymbol{\mu}) := \sum_{i=1}^{n}\boldsymbol{\chi}^i(\mathbf{x})\prod_{j=1}^{m}\omega_j^i(\mu_j). \tag{9}$$

Each mode is composed by the product of a vector-valued function $\boldsymbol{\chi}^i(\mathbf{x})$ representing the spatial displacement and m scalar functions $\omega_j^i(\mu_j)$ representing the influence of the each parameter. For notational simplicity, we will frequently neglect the dependent variables.

To obtain the composing functions, it is natural to compute each mode sequentially from the fact that $\mathbf{u}^i_{\text{PGD}} = \mathbf{u}^{i-1}_{\text{PGD}} + \boldsymbol{\chi}^i(\mathbf{x})\prod_{j=1}^{m}\omega_j^i(\mu_j)$. Such procedure is also called enrichment. For the PGD weak form, we select the admissible variation of the displacement as

$$\mathbf{v} = \mathbf{u}^* := \boldsymbol{\chi}^*\prod_{j=1}^{m}\omega_j + \boldsymbol{\chi}\sum_{k=1}^{m}\omega_k^*\prod_{j=1,j\neq k}^{m}\omega_j, \tag{10}$$

from now on, the index i for current mode and the arguments are frequently omitted for brevity.

To perform the enrichment procedure of finding the ith mode based on the obtained $(i-1)$ modes, we put (9) and (10) into (6). As a result, we have one spatial problem in term of function $\boldsymbol{\chi}$ and m parametric problems in terms of ω_j to be solved. The $(m+1)$ functions can be sought using iterative methods. Typically, the fixed-point alternating directions algorithm is adopted thanks to its simplicity. Consequently, the PGD algorithm contains loops in two levels: one outer loop for modal search and multiple nested loops for solving the $m+1$ functions separately with the others assumed known. During each loop for modal search, by putting (9) and (10) into the weak form (6), we obtain one mechanical problem to solve $\boldsymbol{\chi}$, and m parametric problems to solve ω_j. Take $m=2$ for example, the $m+1=3$ problems read

1. Assuming ω_1 and ω_2 are known, find $\boldsymbol{\chi}$ such that

$$\mathcal{A}(\boldsymbol{\chi}\omega_1\omega_2,\boldsymbol{\chi}^*\omega_1\omega_2) = \mathcal{L}(\boldsymbol{\chi}^*\omega_1\omega_2) - \mathcal{A}(\mathbf{u}^{i-1}_{\text{PGD}},\boldsymbol{\chi}^*\omega_1\omega_2),\quad \forall\boldsymbol{\chi}^*\in V_0. \tag{11}$$

2. Assuming $\boldsymbol{\chi}$ and ω_2 are known, find ω_1 such that

$$\mathcal{A}(\boldsymbol{\chi}\omega_1\omega_2,\boldsymbol{\chi}\omega_1^*\omega_2) = \mathcal{L}(\boldsymbol{\chi}\omega_1^*\omega_2) - \mathcal{A}(\mathbf{u}^{i-1}_{\text{PGD}},\boldsymbol{\chi}\omega_1^*\omega_2),\quad \forall\omega_1^*\in L^2(I_{\mu_1}). \tag{12}$$

3. Assuming $\boldsymbol{\chi}$ and ω_1 are known, find ω_2 such that

$$\mathcal{A}(\boldsymbol{\chi}\omega_1\omega_2,\boldsymbol{\chi}\omega_1\omega_2^*) = \mathcal{L}(\boldsymbol{\chi}\omega_1\omega_2^*) - \mathcal{A}(\mathbf{u}^{i-1}_{\text{PGD}},\boldsymbol{\chi}\omega_1\omega_2^*),\quad \forall\omega_2^*\in L^2(I_{\mu_2}). \tag{13}$$

To implement the algorithm, we define the amplitude of the ith mode for convenience

$$M^i = \|\boldsymbol{\chi}^i\|\prod_{j=1}^{m}\|\omega_j^i\|, \tag{14}$$

where $\|\bullet\|$ denotes the proper norm of $\bullet$ in the corresponding space, which typically is L^2 norm. We present a typical implementation of PGD in Algorithm 1. The main loop is the search for modes, where the maximum number of modes n is assumed given. As we have mentioned before, the

approximation error is measured during the offline computation. In the inside loop which is an iteration indexed with `iter`, the error is controlled *a priori* by a given tolerance `tol`, typically $\texttt{tol} = 10^{-3}$, with stationarity measure of the amplitude as

$$\frac{|M^i_{(\texttt{iter})} - M^i_{(\texttt{iter}-1)}|}{|M^i_{(\texttt{iter}-1)}|} < \texttt{tol}. \tag{15}$$

The update of parametric functions ω^i_j may not necessarily be a for-loop because empirically m is small. Further discussions on PGD algorithm, for instance, the stopping criterion of how to determine the desired number of modes, can be found in the literature such as [4].

Algorithm 1 Typical PGD algorithm

1: Initialize $\boldsymbol{\chi}$ and ω_j.
2: **for** $i = 1$ to n **do**
3: Initialize $\boldsymbol{\chi}^i$ and ω^i_j.
4: **while** $\texttt{iter} < \texttt{iter}_{\max}$ **do**
5: Solve the mechanical problem to update $\boldsymbol{\chi}^i$.
6: **for** $j = 1$ to m **do**
7: Solve the parametric problem to update ω^i_j.
8: **end for**
9: Update the amplitude $M^i_{(\texttt{iter})} \leftarrow \|\boldsymbol{\chi}^i\| \prod_{j=1}^m \|\omega^i_j\|$.
10: Check the convergence:
11: **if** $|M^i_{(\texttt{iter})} - M^i_{(\texttt{iter}-1)}|/|M^i_{(\texttt{iter}-1)}| < \texttt{tol}$ **then**
12: $\texttt{iter} \leftarrow \texttt{iter}_{\max}$
13: **end if**
14: **end while**
15: Save amplitude M^i, functions $\boldsymbol{\chi}^i$ and ω^i_j into vademecum.
16: **end for**

3. NON-INTRUSIVE PGD SCHEME

In this section, we introduce in detail the non-intrusive PGD scheme with the former established mechanical problem in linear elasticity.

From Algorithm 1, we identify the problem as the composition of a sequence of mechanical problems and parametric problems. In each mechanical problem, the costly assembly of the global stiffness matrix is necessary, while for each parametric problem, only the relatively cheaper mass matrix is needed. Therefore, a natural idea is to isolate the mechanical problem, in Line 5 of Algorithm 1 which corresponds to Equation (11), and solve it with some off-the-shelf solvers. For instance, in this paper we implement the scheme with Matlab as the main code and Abaqus as the off-the-shelf solver.

For a better representation, we now introduce the algebraic formulation following standard finite element approximation. In practice, when dealing with structural problems solved through finite element methods, we can express the local mechanical solution $\mathbf{u}(\mathbf{x})$ in terms of nodal degrees of freedom (DOF) $\hat{\mathbf{U}}$ interpolated by the matrix consisting proper shape functions $\mathbf{N}(\mathbf{x})$, i.e., in matrix form, we have $\mathbf{u}(\mathbf{x}) = \mathbf{N}(\mathbf{x})\hat{\mathbf{U}}$. Extending this matrix formulation, we introduce the finite element approximation for both mechanical and parametric problems

$$\boldsymbol{\chi}(\mathbf{x}) = \mathbf{N}(\mathbf{x})\mathbf{U}, \quad \omega_j(\mu_j) = \mathbf{N}^T_j(\mu_j)\boldsymbol{\omega}_j, \tag{16}$$

where $\mathbf{N}_j$ are vectors of shape functions interpolating parameters. Thus (9) becomes

$$\mathbf{u}^n_{\text{PGD}}(\mathbf{x}, \mu_1, \mu_2, \ldots, \mu_m) = \sum_{i=1}^{n} \mathbf{N}(\mathbf{x})\mathbf{U}^i \cdot \prod_{j=1}^{m} \mathbf{N}^T_j(\mu_j)\boldsymbol{\omega}^i_j, \tag{17}$$

where $\mathbf{N}(\mathbf{x})$ denotes the shape functions for the spatial discretization, and $\mathbf{U}^i$ denotes the spatial DOF vector, while $\mathbf{N}_j(\mu_j)$ denotes shape functions for the parameters, and $\boldsymbol{\omega}_j^i$ denotes the DOF vector for the corresponding parameter. Note that $\mathbf{N}(\mathbf{x})$ only depends on the spatial mesh, while $\mathbf{N}_j(\mu_j)$ depends on the discretization of the parametric space. Accordingly, we extend the global nodal displacement vector to

$$\hat{\mathbf{U}}^n_{\text{PGD}}(\mu_1, \mu_2, \dots, \mu_m) := \sum_{i=1}^{n} \mathbf{U}^i \cdot \prod_{j=1}^{m} \mathbf{N}_j^T(\mu_j) \boldsymbol{\omega}_j^i, \tag{18}$$

and similarly we have $\mathbf{u}^n_{\text{PGD}} = \mathbf{N}(\mathbf{x})\hat{\mathbf{U}}^n_{\text{PGD}}$. We would like to address that $\hat{\mathbf{U}}^n_{\text{PGD}}$ is more interested in structural analysis other than the local solution, because it depends only on the parameters, and can be explicitly written as $\hat{\mathbf{U}}^n_{\text{PGD}}(\mu_1, \mu_2, \cdots, \mu_m)$ or $\hat{\mathbf{U}}^n_{\text{PGD}}(\boldsymbol{\mu})$.

3.1. Material properties as extra-coordinates

Consider a separable domain $\Omega = \Omega_1 \cup \Omega_2$ with two different material properties, for example the Young's moduli, as extra-coordinates. For the sake of simplicity, we consider the homogeneous, isotropic materials with the same Poisson's ratio, so that the displacement under given boundary conditions should only depend on the two parameters, E_1 and E_2, of each subdomain. The PGD separated representation reads

$$\hat{\mathbf{U}}^n_{\text{PGD}}(E_1, E_2) := \sum_{i=1}^{n} \mathbf{U}^i \omega_1^i(E_1) \omega_2^i(E_2). \tag{19}$$

In PGD framework, to guarantee the separability of the bilinear form in (6), conventionally it is required that the elastic tensor $\mathbb{C}(\mathbf{x}, E_1, E_2)$ should also be (approximately) separable [24].

In finite element formulation, the elastic tensor $\mathbb{C}(\mathbf{x}, E_1, E_2)$ is discretized to the elemental elastic matrix $\mathbf{D}^{\text{e}}(E_1, E_2)$, and is eventually integrated to the global stiffness matrix $\mathbf{K}(E_1, E_2)$, which now may be written as

$$\mathbf{K}(E_1, E_2) = E_1 \mathbf{K}_1 + E_2 \mathbf{K}_2, \tag{20}$$

where $\mathbf{K}_1$ and $\mathbf{K}_2$ are stiffness-like global matrices, despite that they are independent of Young's moduli. For the convenience, we will keep referring to $\mathbf{K}_1$ and $\mathbf{K}_2$ as stiffness matrices when there is no ambiguity in the context. In the non-intrusive manner, taking advantage of the off-the-shelf codes such as Abaqus, the computation of element stiffness matrices $\mathbf{K}^{\text{e}}$ and the assembly of global stiffness matrix $\mathbf{K}$ is automatically performed with specified values of (E_1, E_2) and the mapping information about their corresponding subdomains. Ideally, we should be able to obtain $\mathbf{K}_1$ by specifying $(E_1, E_2) = (1, 0)$ and $\mathbf{K}_2$ by $(E_1, E_2) = (0, 1)$. However, the limitation from Abaqus that Young's modulus must always be positive prevents this operation. A trick to walk around is to replace zero with a negligible value η to the machine precision, such as $\eta = 10^{-36}$. The stiffness matrices can be stored in files for further use. For Abaqus implementation, the generation and output of global stiffness matrix is requested by adding an extra step in the input file (e.g. `job.inp`) with following commands:

```
*STEP
*MATRIX GENERATE, STIFFNESS
*MATRIX OUTPUT, STIFFNESS, FORMAT=COORDINATE
*END STEP
```

The stiffness matrix is exported by means of two files: a plain text file `job_STIF1.mtx` and a binary file `job_X1.sim`. The data format in the plain text file perfectly matches Matlab format for sparse matrix, so it works as the interface for data exchange from Abaqus to Matlab. The binary file, representing the same matrix, can be reused by Abaqus with a scale factor `sf` (equals to the Young's modulus, for example) through the following commands in any other input file:

```
*MATRIX INPUT, NAME=stiff_1, INPUT=job_X1.sim, MATRIX=STIFFNESS, SCALE FACTOR=sf
*MATRIX ASSEMBLE, STIFFNESS=stiff_1
```

In particular, the discretized form of Equation (11) in this example reads: assuming ω_1, ω_2 and $\hat{\mathbf{U}}^{i-1} = \sum_{k=1}^{i-1} \mathbf{U}^k \omega_1^k \omega_2^k$ are known, solve $\mathbf{U}$ from

$$(\tilde{E}_1 \mathbf{K}_1 + \tilde{E}_2 \mathbf{K}_2)\mathbf{U} = \tilde{Q}\mathbf{F} - \sum_{k=1}^{i-1} (\tilde{E}_1^k \mathbf{K}_1 + \tilde{E}_2^k \mathbf{K}_2)\mathbf{U}^k, \tag{21}$$

where the scalars are computed as follows:

$$\begin{aligned}
&\tilde{E}_1 := \int_{I_{E_1}} E_1 \omega_1^2 \, dE_1 \int_{I_{E_2}} \omega_2^2 \, dE_2, \qquad \tilde{E}_2 := \int_{I_{E_1}} \omega_1^2 \, dE_1 \int_{I_{E_2}} E_2 \omega_2^2 \, dE_2, \qquad \tilde{Q} := \int_{I_{E_1}} \omega_1 \, dE_1 \int_{I_{E_2}} \omega_2 \, dE_2, \\
&\tilde{E}_1^k := \int_{I_{E_1}} E_1 \omega_1 \omega_1^k \, dE_1 \int_{I_{E_2}} \omega_2 \omega_2^k \, dE_2, \quad \tilde{E}_2^k := \int_{I_{E_1}} \omega_1 \omega_1^k \, dE_1 \int_{I_{E_2}} E_2 \omega_2 \omega_2^k \, dE_2, \quad k = 1, \ldots, i-1;
\end{aligned} \tag{22}$$

and $\mathbf{F}$ is the standard finite element load vector, namely

$$\mathbf{F} := \int_\Omega \mathbf{N}^T \mathbf{b} \, d\Omega + \int_{\Gamma_N} \mathbf{N}^T \mathbf{t}_N \, d\Gamma. \tag{23}$$

The load vector can either be read from the Abaqus input file or be written from Matlab data to plain text files and subsequently be included into the Abaqus input file. In this manner, the plain text file establishes the interface for load data transfer.

Note that the only unknown in Equation (21) is $\mathbf{U}$. We denote the right hand side as $\mathbf{F}^*$, and define $\mathbf{K}^* = \tilde{E}_1 \mathbf{K}_1 + \tilde{E}_2 \mathbf{K}_2$. As a result, a fictitious mechanical problem $\mathbf{K}^* \mathbf{U} = \mathbf{F}^*$ is created for Abaqus resolution. In practice, during the construction, only $(\tilde{E}_1, \tilde{E}_2)$ and $\mathbf{F}^*$ need to be computed and output by Matlab through plain text files. There are several ways to obtain $\mathbf{U}$. The simplest one is to require Abaqus directly output the data into a text file (e.g, `job.fil`). One can also use a Python script to extract data from Abaqus output database (e.g, `job.odb`), and write the data into a text file. Here again, this text file works as the interface for transferring the displacement DOFs.

Similarly, the discretized form of the parametric problem Equation (12) reads: assuming $\mathbf{U}$, $\omega_2 = \mathbf{N}_2^T \boldsymbol{\omega}_2$ and $\hat{\mathbf{U}}^{i-1} = \sum_{k=1}^{i-1} \mathbf{U}^k \omega_1^k \omega_2^k$ are known, solve $\omega_1 = \mathbf{N}_1^T \boldsymbol{\omega}_1$ from

$$(\tilde{C}_2 \mathbf{P}_1 + \tilde{M}_2 \mathbf{M}_1)\boldsymbol{\omega}_1 = \tilde{F}_2 \mathbf{Q}_1 - \sum_{k=1}^{i-1} (\tilde{C}_2^k \mathbf{P}_1 + \tilde{M}_2^k \mathbf{M}_1)\boldsymbol{\omega}_1^k \tag{24}$$

where the mass-like matrices on the left hand side are computed by

$$\mathbf{P}_1 := \int_{I_{E_1}} E_1 \mathbf{N}_1 \mathbf{N}_1^T \, dE_1, \quad \mathbf{M}_1 := \int_{I_{E_1}} \mathbf{N}_1 \mathbf{N}_1^T \, dE_1, \tag{25}$$

the load-like vector on the right hand side is computed by

$$\mathbf{Q}_1 := \int_{I_{E_1}} \mathbf{N}_1 \, dE_1, \tag{26}$$

and the scalars are computed by

$$\begin{aligned}
&\tilde{C}_2 := \mathbf{U}^T \mathbf{K}_1 \mathbf{U} \int_{I_{E_2}} \omega_2^2 \, dE_2, \qquad \tilde{M}_2 := \mathbf{U}^T \mathbf{K}_2 \mathbf{U} \int_{I_{E_2}} E_2 \omega_2^2 \, dE_2, \qquad \tilde{F}_2 := \mathbf{U}^T \mathbf{F} \int_{I_{E_2}} \omega_2 \, dE_2, \\
&\tilde{C}_2^k := \mathbf{U}^T \mathbf{K}_1 \mathbf{U}^k \int_{I_{E_2}} \omega_2 \omega_2^k \, dE_2, \quad \tilde{M}_2^k := \mathbf{U}^T \mathbf{K}_2 \mathbf{U}^k \int_{I_{E_2}} E_2 \omega_2 \omega_2^k \, dE_2, \quad k = 1, \ldots, i-1.
\end{aligned} \tag{27}$$

The other parametric problem represented by Equation (13) is solved similarly.

The flowchart of current implementation of the non-intrusive PGD scheme is illustrated in Figure 1. The parametric problems, which are usually computationally inexpensive, are solved within the in-house developed Matlab code, while the fictitious mechanical problems, which are supposed to be much more expensive, are solved by the Abaqus solver called through the following Matlab command:

```
system('abaqus job=job interactive')
```

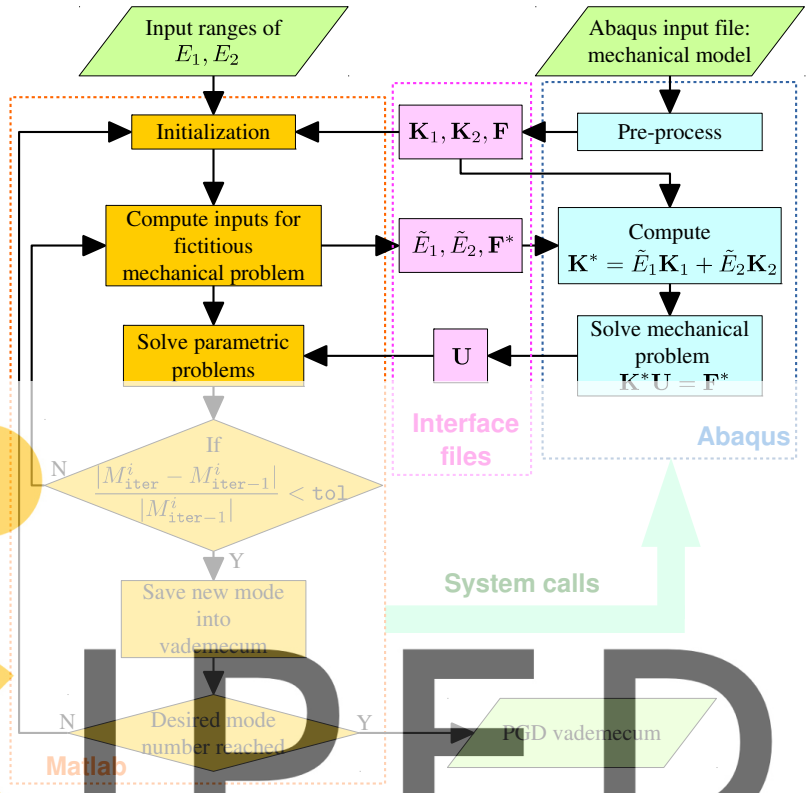


Figure 1. Flowchart of non-intrusive PGD implementation with material properties as extra-coordinates.

An example of Matlab-Abaqus implementation for the non-intrusive PGD scheme can be downloaded through https://github.com/xizou/NIPGD.

3.2. Loading locations as extra-coordinates

In many multi-query problems, we need to obtain solutions under a source term corresponding to different loading locations. Within the PGD framework, the loading location $\mathbf{s} \in \Gamma_N$ is considered as the only extra-coordinate. Considering that the spatial discretization in (16), the global nodal displacement vector (18) can be written as a function of the single parameter $\mathbf{s}$

$$\hat{\mathbf{U}}^n_{\text{PGD}}(\mathbf{s}) := \sum_{i=1}^{n} \mathbf{U}^i \omega^i(\mathbf{s}). \tag{28}$$

We consider the simplest case in problems in linear elasticity that the source term is a point force. With the help of the Dirac delta function, it can be written as a "body force"

$$\mathbf{b}(\mathbf{x}, \mathbf{s}) = \mathbf{b}_0 \cdot \delta(\mathbf{x} - \mathbf{s}), \tag{29}$$

where $\mathbf{b}_0$ is a constant vector that denotes the load, and $\delta(\bullet)$ denotes the Dirac delta function.

Let's focus on the mechanical problem (11). Assuming that the first $i-1$ modes and the ith parametric function $\omega^i(\mathbf{s})$ are known, we now try to solve $\mathbf{U}^i$ for the ith spatial mode. We choose the test function in (10) as $\mathbf{v} = \mathbf{N}(\mathbf{x})\omega^i(\mathbf{s})$, so that the global load vector that corresponding to (23) becomes

$$\mathbf{F}^i(\mathbf{s}) = \int_\Omega \mathbf{N}^T(\mathbf{x})\mathbf{b}_0\omega^i(\mathbf{s}) \cdot \delta(\mathbf{x} - \mathbf{s})\, d\mathbf{x} = \mathbf{N}^T(\mathbf{s})\mathbf{b}_0\omega^i(\mathbf{s}), \tag{30}$$

and its PGD extension becomes

$$\hat{\mathbf{F}}^i = \int_{\Gamma_N} \mathbf{F}^i(\mathbf{s})\, d\mathbf{s} = \int_{\Gamma_N} \mathbf{N}^T(\mathbf{s})\mathbf{b}_0\omega^i(\mathbf{s})\, d\mathbf{s}. \tag{31}$$

Assume the stiffness matrix $\mathbf{K}$ of the system is acquired, the linear elasticity problem can be directly extended to a PGD formulation:

$$\mathbf{K}\hat{\mathbf{U}}^i_{\text{PGD}}(\mathbf{s}) = \hat{\mathbf{F}}^i \Rightarrow \mathbf{K}\sum_{k=1}^{i}\mathbf{U}^k\omega^k(\mathbf{s}) = \int_{\Gamma_N} \mathbf{N}^T(\mathbf{s})\mathbf{b}_0\omega^i(\mathbf{s})\, d\mathbf{s}. \tag{32}$$

Therefore, the spatial DOF $\mathbf{U}^i$ can be obtained by solving equation (32).

Now we need to discretize $\omega^i(\mathbf{s})$ for a numerical solution. A natural approach of discretizing the space of loading location is to follow the spatial FE discretization [17]. That is, we select admissible nodes for the point force from the spatial mesh. The discretization can be written as

$$\omega^i(\mathbf{s}) = \mathbf{N}_{\mathbf{s}}^T(\mathbf{s})\boldsymbol{\omega}^i, \tag{33}$$

where the dimension of the DOF vector $\boldsymbol{\omega}^i$ and that of the corresponding shape function vector $\mathbf{N}_{\mathbf{s}}(\mathbf{s})$ both equal to the number of selected nodes admissible for loading.

Nevertheless, the selected loading locations may not necessarily be the subset of the spatial nodes. In this case, we would like to point out that proper selections of the location of DOF is important for avoiding interpolative instabilities [25].

The implementation of non-intrusive PGD scheme for linear elasticity problems with loading locations as extra-coordinates is similar to 3.1. A detailed example is provided in 5.1.

Remark 1 (Geometrical parameters as extra-coordinates)
In general, geometrical parameters can also be considered as extra-coordinates in the non-intrusive PGD scheme. Due to the geometrical complexity of biological objects, a robust geometrical parametrization framework is still an open question, which is beyond the scope of this paper. Focusing on the community of reduced order modelling, the interesting approaches that the authors would like to mention include the level-set based method for treating inclusions investigated in [26] and the non-uniform rational B-spline (NURBS) parameterization discussed in [27].

4. PGD WITH CONSTRAINED PARAMETRIC SPACE

In typical PGD frameworks, the parameters are assumed to be independent to each other, and the parametric space is assumed to be Cartesian, i.e., $I_{\mu_1} \times \cdots \times I_{\mu_m}$, so that the parametric dependent part in the separated representation can be written in terms of a product of m 1D functions. Thanks to such an assumption, the parametric problems to be solved in the alternated directions during each modal search are all 1D, which enables a fast computing process.

On the contrary, in real situations, the parameters that are not independent mutually causes the parametric space not being Cartesian due to the constrains from limitations of physical, geometrical, and/or other aspects. The constrained parametric space has much lower separability, which could cause the separated representation ineffective. That is, the factorization (8) may not be able to perform. To this problem, one possible solution is to apply the Singular Value Decomposition (SVD). In this section, we propose another solution that collects the mutually dependent parameters and separates the parametric space into 2D or 3D spaces. Parameters living in the 2D or 3D spaces are always kept unseparated.

A common situation is that the parametric space can be 2D or 3D when representing the loading locations. In this case, the parametric space is not separable in the Cartesian fashion. Instead of separating it into tensor product of 1D parametric spaces, we prefer to keep it unseparated and solve the parametric problem in a 2D or 3D space. In this context, the PGD separated representation (9)

can be rewritten as

$$\mathbf{u}(\mathbf{x},\boldsymbol{\mu}) = \mathbf{u}^n_{\text{PGD}} := \sum_{i=1}^{n} \boldsymbol{\chi}^i(\mathbf{x})\omega^i(\boldsymbol{\mu}). \tag{34}$$

Let's consider another case in which the correlated material parameters forms a constrained parametric space. We use again the expression from 3.1, in which the constrained parametric space is assumed 2D, namely $\Omega_{\text{c}} \subset \mathbb{R}^2$. Let $\boldsymbol{\mu} = (E_1, E_2) \in \Omega_{\text{c}}$, now (21) becomes: assuming $\omega(\boldsymbol{\mu})$ and $\hat{\mathbf{U}}^{i-1} = \sum_{k=1}^{i-1} \mathbf{U}^k\omega^k(\boldsymbol{\mu})$ are known, solve $\mathbf{U}$ from

$$(\tilde{E}_1\mathbf{K}_1 + \tilde{E}_2\mathbf{K}_2)\mathbf{U} = \tilde{Q}\mathbf{F} - \sum_{k=1}^{i-1}(\tilde{E}_1^k\mathbf{K}_1 + \tilde{E}_2^k\mathbf{K}_2)\mathbf{U}^k, \tag{35}$$

where the scalars are computed as follows:

$$\begin{aligned} \tilde{E}_1 &:= \int_{\Omega_{\text{c}}} E_1\omega^2(\boldsymbol{\mu})\,d\boldsymbol{\mu}, \qquad \tilde{E}_2 := \int_{\Omega_{\text{c}}} E_2\omega^2(\boldsymbol{\mu})\,d\boldsymbol{\mu}, \qquad \tilde{Q} := \int_{\Omega_{\text{c}}} \omega(\boldsymbol{\mu})\,d\boldsymbol{\mu}, \\ \tilde{E}_1^k &:= \int_{\Omega_{\text{c}}} E_1\omega(\boldsymbol{\mu})\omega^k(\boldsymbol{\mu})\,d\boldsymbol{\mu}, \quad \tilde{E}_2^k := \int_{\Omega_{\text{c}}} E_1\omega(\boldsymbol{\mu})\omega^k(\boldsymbol{\mu})\,d\boldsymbol{\mu}, \quad k = 1,\ldots,i-1. \end{aligned} \tag{36}$$

The integrals are computed repeatedly during the loops, so additional computational costs may be introduced since $\dim\Omega_{\text{c}} = 2$. On the other hand, we now have more options to discretize the parametric space with different meshes. For example, one may use h-refinement to improve the accuracy.

When the parameters $\boldsymbol{\mu}$ are limited in a constrained space Ω_{c}, such a technique would lead to the total non-intrusive PGD scheme described in Algorithm 2. In this fully non-intrusive scheme, the main code acts only as the data collector and flow controller, while the computation of equations are performed by using the exterior solver as a black box.

Algorithm 2 Fully non-intrusive PGD algorithm

1: Create spatial mesh and parametric mesh for the model.
2: Transfer the definition of the model into main code.
3: Initialize $\boldsymbol{\chi}(\mathbf{x})$ and $\omega(\boldsymbol{\mu})$.
4: **for** $i = 1$ to n **do**
5: Initialize $\boldsymbol{\chi}^i(\mathbf{x})$ and $\omega^i(\boldsymbol{\mu})$.
6: **while** $\texttt{iter} < \texttt{iter}_{\max}$ **do**
7: Export and solve the mechanical problem in external solver.
8: Import the solution into main code to update $\boldsymbol{\chi}^i(\mathbf{x})$.
9: Export and solve the parametric problem in external solver.
10: Import the solution into main code to update $\omega^i(\boldsymbol{\mu})$.
11: Update the amplitude $M^i_{(\texttt{iter})} \leftarrow \|\boldsymbol{\chi}^i(\mathbf{x})\| \cdot \|\omega^i(\boldsymbol{\mu})\|$.
12: Check the convergence:
13: **if** $|M^i_{(\texttt{iter})} - M^i_{(\texttt{iter}-1)}|/|M^i_{(\texttt{iter}-1)}| < \texttt{tol}$ **then**
14: $\texttt{iter} \leftarrow \texttt{iter}_{\max}$
15: **end if**
16: **end while**
17: Save functions $\boldsymbol{\chi}^i(\mathbf{x})$ and $\omega^i(\boldsymbol{\mu})$ into vademecum.
18: **end for**

Remark 2 (On the solving of parametric problems)
The parametric problems, as represented in Line 9 in Algorithm 2, does not contain derivatives with respect to the unknowns, which suggests that it is not always necessary to solve the parametric problems with the Galerkin-based FE method.

In the non-intrusive implementation of the PGD algorithm, each call of the external solver will introduce extra cost of computation time, because upon each call, the external solver (e.g. Abaqus) will perform its initialization.

In this paper, the parametric problems for all the examples are solved within the in-house Matlab code still with FE discretization, despite that they can also be solved by calling Abaqus.

5. NUMERICAL EXAMPLES

5.1. Example I: A 1D model problem with loading locations as extra-coordinates

In this simple example, we present an instance of non-intrusive PGD implementation with loading location as extra-coordinates, and compare the ways of obtaining the reduced order solution.

As depicted in Figure 2, consider an elastic bar with left end Point 1 ($x = 0$) fixed and right end Point 3 ($x = l$) free, and loaded by F at Point 2 ($x = s$ with $s \in [s_1, s_2]$). The displacement u is a function of the querying location x and the loading location s, i.e., $u = u(x; s)$. The strong form of the problem reads

$$\begin{cases} \dfrac{d}{dx}\left(EA\dfrac{du}{dx}\right) + f = 0, & \text{in } \Omega = (0, l), \\ u(0) = 0, \qquad EA\dfrac{du}{dx}(l) = 0, \\ f = F\delta(x - s). \end{cases} \tag{37}$$

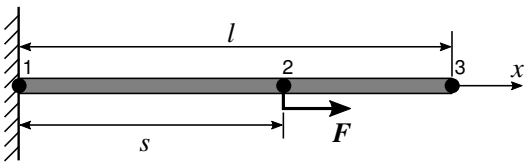


Figure 2. Sketch of the 1D model problem with loading location as extra-coordinates.

The exact solution to problem (37) is:

$$u(x, s) = \begin{cases} \dfrac{Fx}{EA}, & \text{if } x < s, \\ \dfrac{Fs}{EA}, & \text{if } x \geq s. \end{cases} \tag{38}$$

The PGD separated representation reads

$$u(x, s) = u^n_{\text{PGD}} = \sum_{i=1}^{n} U^i(x)\omega^i(s). \tag{39}$$

The extended weak form reads

$$\int_{s_1}^{s_2}\int_0^l EA\frac{dv}{dx}\frac{du}{dx}\,dx\,ds = \int_{s_1}^{s_2}\int_0^l vf\,dx\,ds, \tag{40}$$

where $v = u^*$ is the variation. Discretize the functions as $U^i(x) = \mathbf{N}_x^T(x)\mathbf{U}_i$ and $\omega^i(s) = \mathbf{N}_s^T(s)\mathbf{S}_i$, we obtain the algebraic form according to the variation. Note we select the admissible loading locations from the discretized spatial nodes, so shape functions for loading locations $\mathbf{N}_s(s)$ is a subset of spatial shape functions $\mathbf{N}_x(x)$. Let $v = \mathbf{N}_x(x) \cdot \mathbf{N}_s^T(s)\mathbf{S}_i$, the right hand side becomes

$$\int_{s_1}^{s_2}\int_0^l \mathbf{N}_x(x) \cdot \mathbf{N}_s^T(s)\mathbf{S}_i F\delta(x - s)\,dx\,ds = F\left[\int_{s_1}^{s_2} \mathbf{N}_x(s)\mathbf{N}_s^T(s)\,ds\right]\mathbf{S}_i. \tag{41}$$

In the last term, the integral on the parametric space $[s_1, s_2]$ is a mass-like matrix, however, as discussed in 3.2, here it is not necessarily a squared matrix because it involves the shape functions of both spatial and parametric discretization, which may have different dimensions. This matrix has to be computed intrusively in the main code. We denote it as

$$\tilde{\mathbf{M}} = \int_{s_1}^{s_2} \mathbf{N}_x(s)\mathbf{N}_s^T(s)\, ds. \tag{42}$$

Regarding the left hand side, it reads

$$\sum_{i=1}^{n} \mathbf{S}_i^T \left(\int_{s_1}^{s_2} \mathbf{N}_s \mathbf{N}_s^T \, ds \right) \mathbf{S}_i \cdot \left(\int_0^l EA \frac{d\mathbf{N}_x}{dx} \frac{d\mathbf{N}_x^T}{dx} \, dx \right) \mathbf{U}_i. \tag{43}$$

The mass matrix in parametric space $\mathbf{M}$ and stiffness matrix for the mechanical problem $\mathbf{K}$ in the parentheses can be identified:

$$\mathbf{M} = \int_{s_1}^{s_2} \mathbf{N}_s \mathbf{N}_s^T \, ds, \quad \mathbf{K} = \int_0^l EA \frac{d\mathbf{N}_x}{dx} \frac{d\mathbf{N}_x^T}{dx} \, dx. \tag{44}$$

The mechanical problem in PGD modal search which equivalent to (11) now reads: assuming $\mathbf{S}_i$ and $\hat{\mathbf{U}}^{i-1} = \sum_{k=1}^{i-1} \mathbf{U}_k \cdot \mathbf{N}_s^T \mathbf{S}_k$ are known, solve $\mathbf{U}_i$ from

$$(\mathbf{S}_i^T \mathbf{M} \mathbf{S}_i) \cdot \mathbf{K}\mathbf{U}_i = F\tilde{\mathbf{M}}\mathbf{S}_i - \sum_{k=1}^{i-1} (\mathbf{S}_i^T \mathbf{M} \mathbf{S}_k) \cdot \mathbf{K}\mathbf{U}_k. \tag{45}$$

For this particular problem, we set $l = 100$, $s_1 = 50$ and $s_2 = 75$. The bar is discretized into a 100-element mesh, from which 25 elements are extracted to construct the parametric space, so the non-squared mass matrix $\tilde{\mathbf{M}}$ has a dimension of 101×26. The extended parametric problem is 2D, according to [4], both SVD and PGD approach can be used to obtain the reduced basis. We may construct a matrix by concatenating the n solutions as columns, while the number of rows depends on the degrees of freedom in the problem. It is optimal to perform singular value decomposition (SVD) on the matrix to obtain the reduced order solution. Meanwhile, one may also obtain the suboptimal reduced order solution with PGD. In Figure 3, we illustrate both results with the decreasing curve of amplitude versus modes. It demonstrates that the PGD solution is close to the SVD solution, and is suboptimal than the SVD solution which is optimal. The drastic drops occurring both after the 26th mode indicate that the real order of solution for this model problem is 26, which agrees with the discretization of the parametric space. The comparison between PGD solution and the exact solution is shown in Figure 4. With 26 PGD modes, the maximum relative error is 0.27%.

Remark 3 (Non-intrusive computation for the matrices)
Note that the mass and stiffness matrices need only compute once for all. It is possible to use Abaqus to compute the stiffness matrix. In principle, it should also be possible to use Abaqus to compute the mass matrix for the discretized parametric space, however, Abaqus can only output lumped mass matrix for linear elements. To obtain consistent mass matrix, we have to compute it with in-house codes. Besides, the non-squared mass matrix has to be computed with in-house codes.

5.2. Example II: A 2D model problem with material properties as extra-coordinates

Consider a plane strain problem in a squared spatial domain $[0, 10] \times [0, 10]$, as sketched in Figure 5, the left edge and lower edge are fixed horizontally and vertically, respectively, while the upper edge is free, and the right edge is loaded with uniform traction σ. The domain is partitioned into two subdomain composed by materials with the same Poisson's ratio $\nu = 0.3$ but different Young's moduli E_1 and E_2. Provided having used a compatible unit system, we will ignore units in this example.

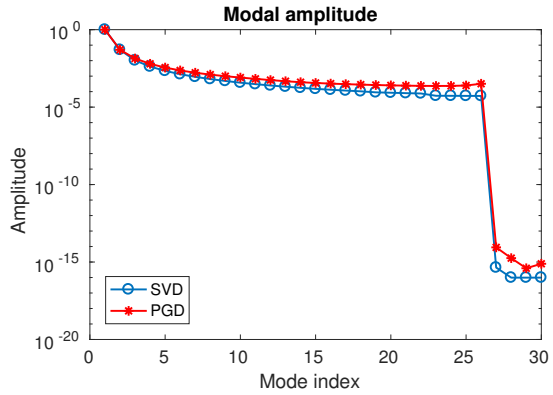


Figure 3. Comparison of PGD and SVD solutions. The modal amplitudes drastically drop after the 26th mode.

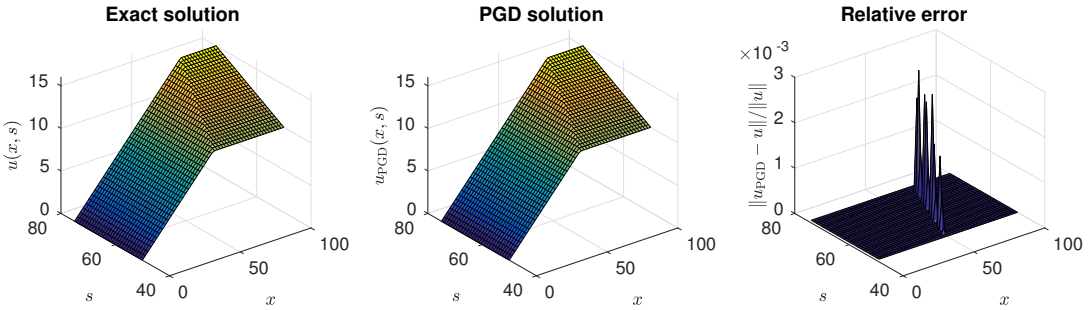


Figure 4. Comparison of the exact solution and PGD solution with 26 modes.

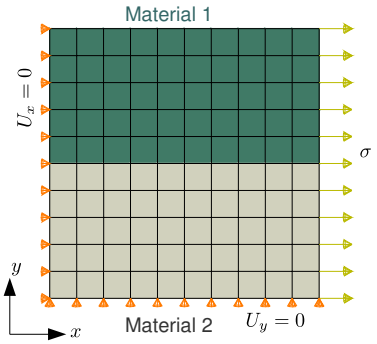


Figure 5. Sketch of the 2D model problem with material properties as extra-coordinates.

For the purpose of testing the new strategy of separating the parametric space detailed in 4, we implement the non-intrusive PGD scheme introduced in 3.1 to solve the problem twice with parametric space separated and unseparated, namely

$$\hat{\mathbf{U}}^n_{\text{PGD}} = \sum_{i=1}^{n} \mathbf{U}^i_{\text{sep}} \omega^i_1(E_1)\omega^i_2(E_2) = \sum_{j=1}^{n} \mathbf{U}^j_{\text{unsep}} \omega^j(E_1, E_2). \tag{46}$$

Note that although the spatial domain is identical, the spatial modes result from separated approach $\mathbf{U}^i_{\text{sep}}$ and from unseparated $\mathbf{U}^j_{\text{unsep}}$ are not the same, neither are the functions of parameters.

The ranges of parameters are selected as $E_1 \in [10, 100], E_2 \in [20, 200]$. For the separated approach, the parametric space is $[10, 100] \otimes [20, 200]$, and 100 linear elements are used for the discretization both of E_1 and E_2. For the unseparated approach, the parametric space is

$[10, 100] \times [20, 200]$. We used three different linear triangular meshes to discretize the unseparated parametric space: one with 114 unstructured triangles, one with 100 structured triangles, and to be more comparable with the separated approach, the last one with $100 \times 100 \times 2 = 20000$ triangles. All the meshes are shown in Figure 11. We computed 10 modes for both approaches of discretizing the parametric space with the same stopping criterion, $\texttt{tol} = 10^{-3}$, for the looping control.

The amplitude of each mode is plotted in Figure 6. It demonstrates that the unseparated approach achieves higher rate of convergence than the conventional separated PGD approach. The results imply that the unseparated approach, although presumed to be more expensive than the separated approach, may be not so expensive because less modes are needed to achieve the same accuracy thanks to the higher rate of convergence. It is worth noting that some authors such as [1] also compared the separated approach and unseparated approach in the spatial domain. Similarly, the unseparated approach exhibits a higher rate of convergence.

The spatial modes for both approaches are plot in Figure 7 and Figure 8 with nodal displacement magnitudes $U = \sqrt{U_x^2 + U_y^2}$. To visualize the correlation between the two parameters, we plot the tensor product of results from the separated approach in Figure 9 and the results from the unseparated approach along with the mesh in Figure 10. It is observed that the first three modes, either spatial or parametric, computed by both separation strategies are very similar.

To evaluate the error for both approaches, we have randomly selected 5 samples of parameters (E_1, E_2) and computed $\mathbf{U}_{\text{ref}}(E_1, E_2)$ with corresponding ordinary FE models as references. The relative error $\|\hat{\mathbf{U}}_{\text{PGD}} - \mathbf{U}_{\text{ref}}\|/\|\mathbf{U}_{\text{ref}}\|$ for each sample for both approaches are plotted in Figure 11. As we have observed, the unseparated approach converges more rapidly. The refinement of mesh for the unseparated parametric space improves significantly the accuracy of the converged result, but it requires more modes to achieve the convergence. Take the sample $(E_1, E_2) = (76.24, 29.86)$ for example, the relative error converges near 10^{-2} in the coarse meshes with about 100 triangles and the convergence requires 4–5 modes, while it converges near 10^{-4} in the fine mesh with 20000 triangles and the convergence requires 8 modes.

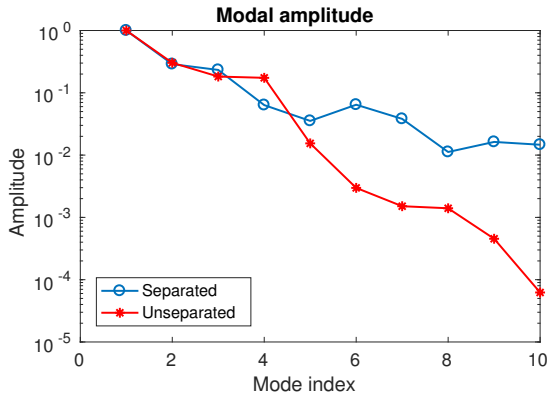


Figure 6. Normalized modal amplitude plots for both separated and unseparated approach.

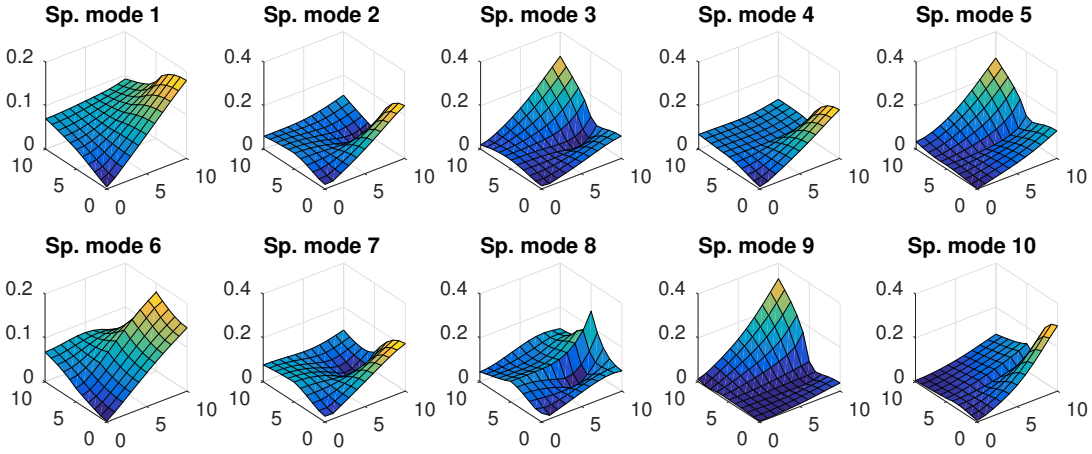


Figure 7. Normalized spatial modes represented by magnitude of $\mathbf{U}^i_{\text{sep}}$ solved from separated parametric space. Bottom plane: spatial domain. Vertical axis: magnitude of nodal displacement.

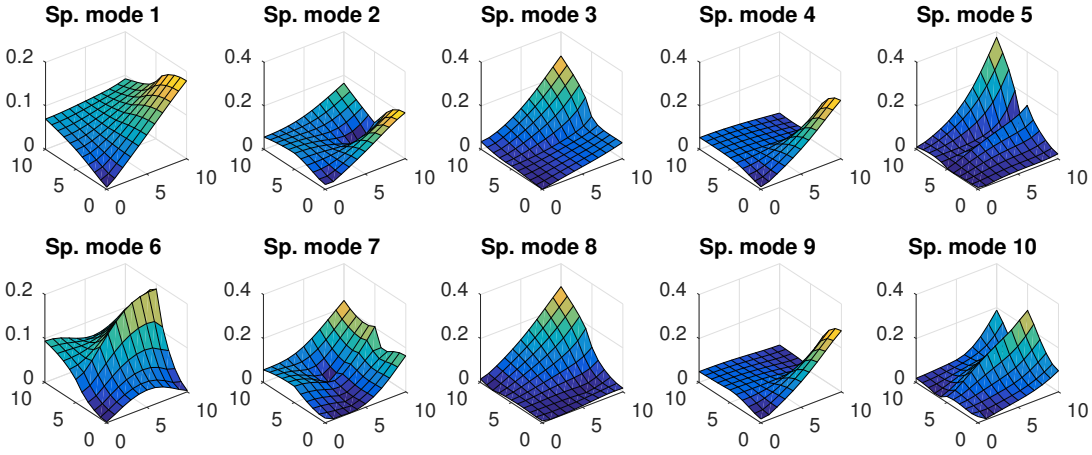


Figure 8. Normalized spatial modes represented by magnitude of $\mathbf{U}^i_{\text{unsep}}$ solved from unseparated parametric space. Bottom plane: spatial domain. Vertical axis: magnitude of nodal displacement.

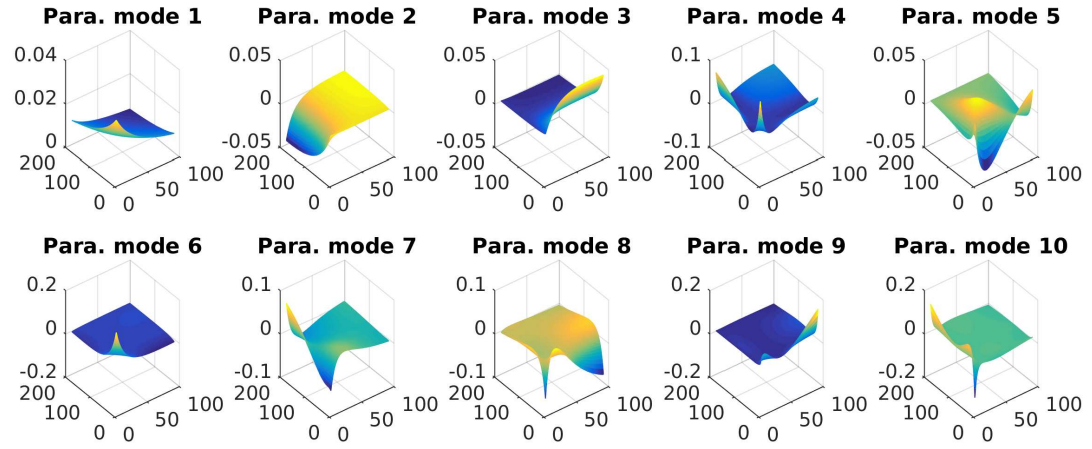


Figure 9. Normalized parametric modes $\omega^j_1(E_1) \otimes \omega^j_2(E_2)$ solved from separated parametric space. Bottom plane: parametric domain. Vertical axis: value of the parametric function.

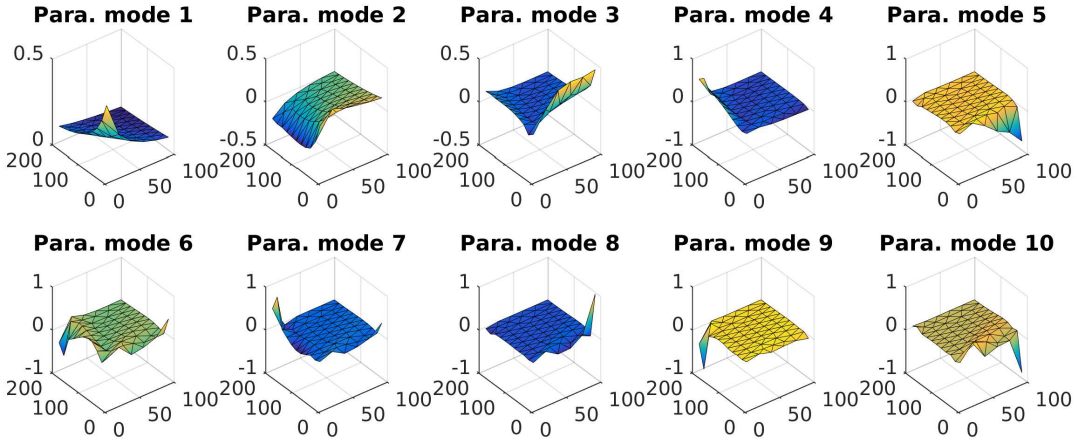
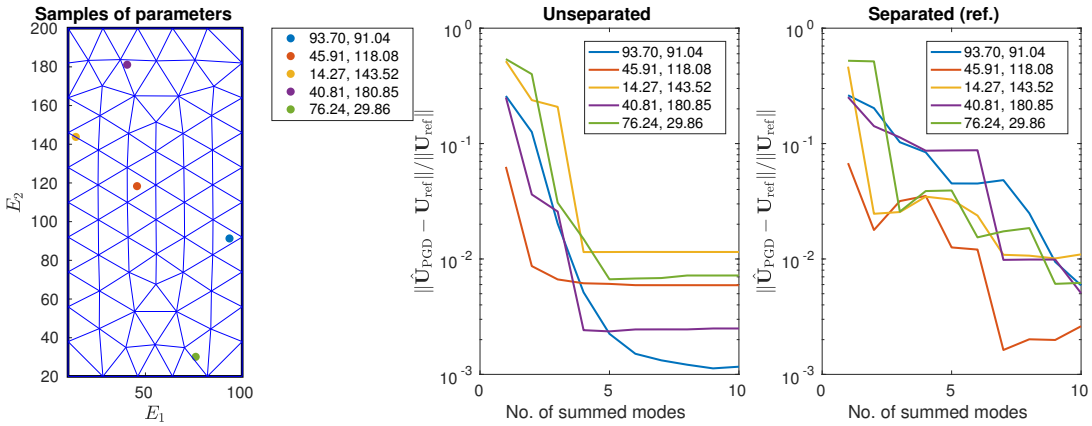
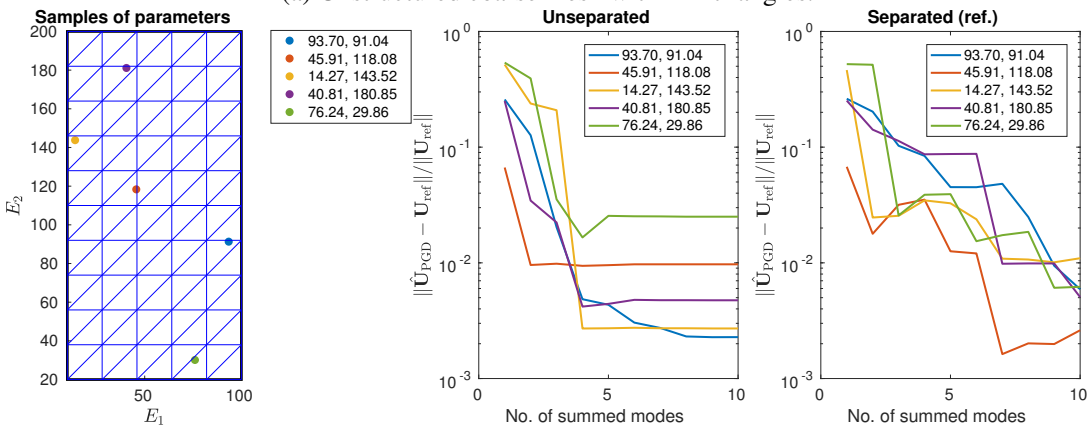


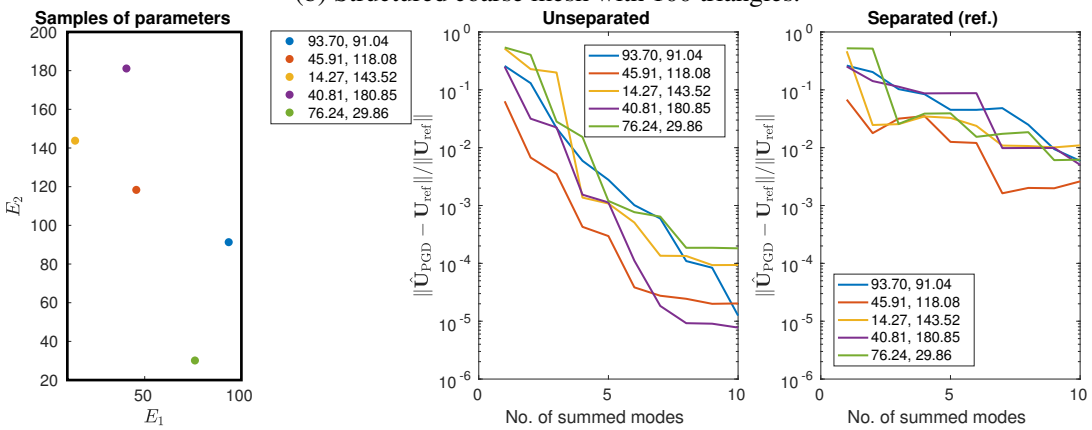
Figure 10. Normalized parametric modes $\omega^j(E_1, E_2)$ solved from unseparated parametric space. Bottom plane: parametric domain. Vertical axis: value of the parametric function.



(a) Unstructured coarse mesh with 114 triangles.



(b) Structured coarse mesh with 100 triangles.



(c) Structured fine mesh with 20000 triangles (mesh not shown).

Figure 11. Relative error of both separated and unseparated PGD approach for the 2D model problem. For the separated approach, three different meshes for the parametric space are used. For the unseparated approach, the discretization remains the same.

6. APPLICATION IN BIOMECHANICAL PROBLEMS: HUMAN PROXIMAL FEMUR

Literature [19, 20, 21, 22, 28, 26] shows that the PGD framework is particularly suitable for many challenging problems in biomechanics such as real-time simulation and multi-parameter identification. In a comprehensive ageing society, osteoporosis is one of the common diseases for the elders. Total hip arthroplasty (THA) [29] is a typical surgery for osteoporosis happened on the proximal femur. A patient-specific, fast-responding decision making tool could be of great interest for surgeons facing the THA to study the mechanical response of the proximal femur.

The hypothesis of linear elastic modelling of bone tissue at macroscale is commonly accepted for biomechanical analysis under normal loads in regular daily activities [30, 31].

In this section, we present a real problem in biomechanics, and apply the scheme proposed to demonstrate its feasibility. We have previously performed CT scans as well as experimental tests to obtain data from a real sample. In biomechanics, the macro structure of bone tissue is commonly assumed to be linear elastic, so the numerical scheme well suits the application.

6.1. CT image-based modelling and intensity-modulus mapping

The CT images represents a 3D space of a box shape, i.e., $\Omega_{\mathrm{CT}} = [0, a] \otimes [0, b] \otimes [0, c]$, where a, b, c denote the length, width and height of the box, respectively. Each CT image is a slice with $n_a \times n_b$ pixels, the resolution r is determined by the CT machine, while $n_a = a/r$ and $n_b = b/r$. The distance between slices $\Delta_c = c/n_c$ can be controlled for different type of scans, resulting n_c slices. In total, a CT image set contains $n_a \times n_b \times n_c$ voxels and their corresponding grayscale value. Each voxel grayscale value is mapped to the intensity ρ, which represents the density of the scanned location.

The first step is to extract the interested domain Ω from Ω_{CT}, this segmentation procedure filters the voxels that are void in the box, leaving only the part with $\rho > 0$. In particular, through the procedure we have obtained $\rho \in [1, 3071]$, with ρ being integer.

The spatial mesh for the finite element model of proximal femur is shown in Figure 12. All the elements are linear tetrahedral element (C3D4 provided by Abaqus). The model is fixed at the distal end, and loaded on the femur head in vertical direction. To simulate the strain acquisition of the experiments, the numerical strains are extracted from the approximate positions located on two sections on the shaft.

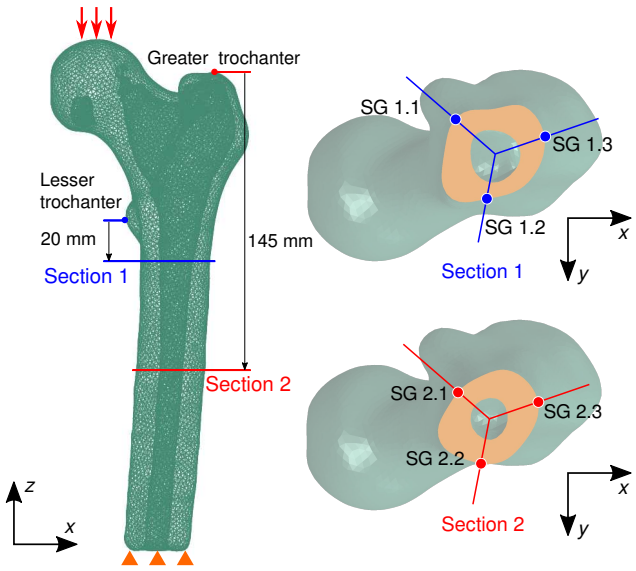


Figure 12. Spatial mesh of the human proximal femur model, illustrating locations with strain gauges attached for strain acquisition. Each location is attached two strain gauges in longitudinal and circumferential direction, respectively.

According to literature [32, 33] that focuses on the simulation of bone mechanics with structural elements, we adopt the simplest assumption that the intensity of CT images is linearly mapped to the Young's modulus of the bone:

$$E(\mathbf{x}) = \alpha\rho(\mathbf{x}) + \beta, \tag{47}$$

where α and β are parameters that can be determined by experiments. With this mapping, we ignore the difference in biological tissues between the trabecular bone and cortical bone, but use variable moduli to represent its inhomogeneity in mechanics. Note that the CT image is, in fact, already discretized. For simplicity, in the finite element model we assign each element the intensity from the voxel which is closest to the centroid of the element, denoted as ρ^{e}.

6.2. PGD separated representation for Young's modulus mapped from CT image

In PGD framework, the Young's modulus is again extended from $E(\mathbf{x})$ to $E(\mathbf{x}, \alpha, \beta)$. Thanks to the linear mapping, Young's modulus is separable in this form:

$$E(\mathbf{x}, \alpha, \beta) = \sum_{k=1}^{2} G_k(\mathbf{x}) R_k(\alpha) S_k(\beta), \tag{48}$$

where

$$\begin{aligned} &R_1(\alpha) = \alpha, \quad S_1(\beta) = 1, \quad G_1(\mathbf{x}) = \rho(\mathbf{x}), \\ &R_2(\alpha) = 1, \quad S_2(\beta) = \beta, \quad G_2(\mathbf{x}) = 1. \end{aligned} \tag{49}$$

In standard finite element methods, the element stiffness matrix $\mathbf{K}^{\mathrm{e}}$ is computed as

$$\mathbf{K}^{\mathrm{e}} := \int_{\Omega} \mathbf{B}^T \mathbf{D}^{\mathrm{e}} \mathbf{B} \, d\Omega = \int_{\Omega} E^{\mathrm{e}} \cdot \mathbf{B}^T \hat{\mathbf{D}} \mathbf{B} \, d\Omega = \int_{\Omega} (\alpha\rho^{\mathrm{e}} + \beta) \mathbf{B}^T \hat{\mathbf{D}} \mathbf{B} \, d\Omega, \tag{50}$$

where $\mathbf{B} := \nabla^{\mathrm{s}} \mathbf{N}$ is the displacement-strain matrix, $\mathbf{D}^{\mathrm{e}}$ is the elemental elastic matrix:

$$\mathbf{D}^{\mathrm{e}} := \frac{E^{\mathrm{e}}}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & (1-2\nu)/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & (1-2\nu)/2 & 0 \\ 0 & 0 & 0 & 0 & 0 & (1-2\nu)/2 \end{bmatrix}. \tag{51}$$

We define that $\hat{\mathbf{D}} := \mathbf{D}^{\mathrm{e}}/E^{\mathrm{e}}$. In linear elastic case with given Poisson's ratio ν, $\hat{\mathbf{D}}$ is constant. Now the the element stiffness matrix is separated as

$$\mathbf{K}^{\mathrm{e}} = \alpha \mathbf{K}_1^{\mathrm{e}} + \beta \mathbf{K}_2^{\mathrm{e}}, \tag{52}$$

with

$$\mathbf{K}_1^{\mathrm{e}} := \int_{\Omega} \rho^{\mathrm{e}} \cdot \mathbf{B}^T \hat{\mathbf{D}} \mathbf{B} \, d\Omega, \quad \mathbf{K}_2^{\mathrm{e}} := \int_{\Omega} \mathbf{B}^T \hat{\mathbf{D}} \mathbf{B} \, d\Omega. \tag{53}$$

It is clear when assembled into the global stiffness matrix, $\mathbf{K}_1^{\mathrm{e}}$ contributes to the inhomogeneous part, while $\mathbf{K}_2^{\mathrm{e}}$ consists the homogeneous part.

Note the similarity between (52) and (20), the computation procedure discussed in 3.1 well suits this problem. However, due to the CT imaging mechanism that stronger tissue results higher intensity, apparently we have $\alpha > 0$. In addition, physically Young's modulus cannot be negative, the parameters α and β must satisfy following conditions:

$$\begin{cases} \alpha > 0, \\ \alpha + \beta > 0. \end{cases} \tag{54}$$

Therefore, we have obtained a problem with parameters $\boldsymbol{\mu} = (\alpha, \beta)$ living in a constrained 2D parametric domain Ω_{c}, as shown in Figure 13. The unseparated approach (35) applies.

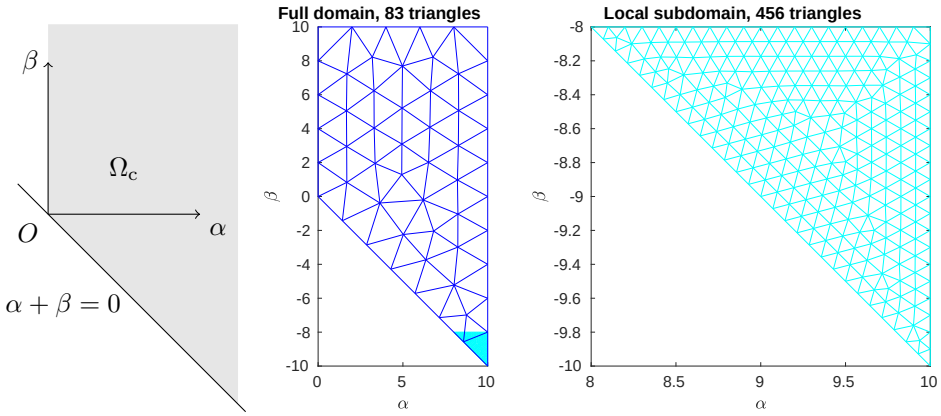


Figure 13. Discretization of the constrained parametric domain. Left: constrained parametric space; middle: coarse mesh with 83 triangles, $(\alpha_{\max}, \beta_{\max}) = (10, 10)$; right: subdomain mesh with 456 triangles, $(\alpha_{\max}, \beta_{\max}) = (10, -8)$.

In practice, we confine the parametric domain by proper choices of $(\alpha_{\max}, \beta_{\max})$, and discretize the domain with triangular mesh. In the parameter identification problem described in 6.3, since the values of (α^*, β^*) is unknown while the parametric mesh is flexible, we use a coarser mesh to perform preliminary estimation, and subsequently use a finer mesh to approximate the desired values.

The separated representation for the nodal displacement vector reads

$$\hat{\mathbf{U}}_{\text{PGD}}(\boldsymbol{\mu}) := \sum_{i=1}^{n} \mathbf{U}^i \omega^i(\boldsymbol{\mu}). \tag{55}$$

Applying the non-intrusive PGD scheme with different mesh of the constrained parametric space Ω_c, we are able to obtained the corresponding computational vademecum $\hat{\mathbf{U}}_{\text{PGD}}(\boldsymbol{\mu})$.

6.3. Parameter identification with experimental data

The real parameters $\boldsymbol{\mu}^* = (\alpha^*, \beta^*)$ for the model are unknown *a priori*. We have performed *in vitro* experiments on the femur, and have obtained data from the 12 strain gauges depicted in Figure 12, namely $\boldsymbol{\varepsilon}^*_{\mathbb{P}} = [\epsilon_1, \epsilon_2, \ldots, \epsilon_{12}]^T$. Taking advantage of the power of PGD vademecum [34], we now construct a parameter identification problem to obtain the parameters.

The numerical strain is computed as

$$\boldsymbol{\varepsilon} = \nabla^{\text{s}}\mathbf{u} = \nabla^{\text{s}}\mathbf{N}\hat{\mathbf{U}} = \mathbf{B}\hat{\mathbf{U}}. \tag{56}$$

Regarding to PGD separated representation, when the displacement vademecum $\hat{\mathbf{U}}_{\text{PGD}}(\boldsymbol{\mu})$ is already obtained, the strain vademecum can be computed from

$$\boldsymbol{\varepsilon}_{\text{PGD}}(\boldsymbol{\mu}) = \mathbf{B}\hat{\mathbf{U}}_{\text{PGD}}(\boldsymbol{\mu}) = \mathbf{B}\sum_i \mathbf{U}^i\omega^i(\boldsymbol{\mu}) = \sum_i \mathbf{B}\mathbf{U}^i\omega^i(\boldsymbol{\mu}). \tag{57}$$

Defining a selection operator $\mathbb{P}$ to extract the 12 numerical strain values corresponding to the experiment result, the selected strain can be written as a vector

$$\boldsymbol{\varepsilon}_{\mathbb{P}}(\boldsymbol{\mu}) = \mathbb{P}\boldsymbol{\varepsilon}_{\text{PGD}}(\boldsymbol{\mu}) = \mathbb{P}\mathbf{B}\sum_i \mathbf{U}^i\omega^i(\boldsymbol{\mu}) = \sum_i \mathbb{P}\mathbf{B}\mathbf{U}^i\omega^i(\boldsymbol{\mu}). \tag{58}$$

The parameter identification problem reads: given $\boldsymbol{\varepsilon}_{\text{PGD}}(\boldsymbol{\mu})$, find $\boldsymbol{\mu}^*$ such that

$$\boldsymbol{\mu}^* = \underset{\boldsymbol{\mu}\in\Omega_c}{\arg\min} \|\boldsymbol{\varepsilon}^*_{\mathbb{P}} - \boldsymbol{\varepsilon}_{\mathbb{P}}(\boldsymbol{\mu})\|. \tag{59}$$

To certify the eligibility of the parameter identification problem, tests have been made with synthetic values for the parameters. We have computed $\boldsymbol{\varepsilon}^*_{\mathbb{P}}$ with synthetic parameters $\boldsymbol{\mu}^* = (3.02, 3680)$, and then we solved the identification problem with $\boldsymbol{\varepsilon}^*_{\mathbb{P}}$ and the previously computed vademecum $\boldsymbol{\varepsilon}_{\mathbb{P}}(\boldsymbol{\mu})$. The result $\boldsymbol{\mu} = (3.0168, 3667.22)$ with relative error $\|\boldsymbol{\mu} - \boldsymbol{\mu}^*\|/\|\boldsymbol{\mu}^*\| = 0.35\%$ shows a good agreement which proves the identification.

For real case with $\boldsymbol{\varepsilon}^*_{\mathbb{P}}$ from the experiments, the result of the parameter identification problem is shown in Figure 14. With $\boldsymbol{\varepsilon}_{\mathbb{P}}(\boldsymbol{\mu})$ computed from the coarse parametric mesh, the approximate location of the parameters is identified as $\boldsymbol{\mu}^* = (9.3712, -9.3712)$. Thereafter, the vademecum is recomputed with the fine parametric mesh, and the parameters are identified as $\boldsymbol{\mu}^* = (9.3589, -9.3589)$.

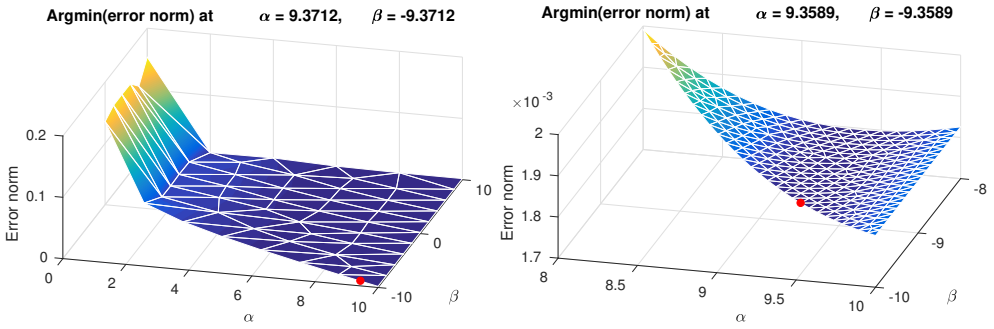


Figure 14. Error norm $\|\boldsymbol{\varepsilon}^*_{\mathbb{P}} - \boldsymbol{\varepsilon}_{\mathbb{P}}(\alpha, \beta)\|$ distribution on the parametric space. Left: preliminary result from coarse mesh; right: secondary result fine mesh on local subdomain.

Remark 4 (Computation of numerical strains)
In practice, there are two options for the implementation of (58). One option is compute $\mathbf{B}$ within the main code after having obtained $\hat{\mathbf{U}}_{\text{PGD}}(\boldsymbol{\mu})$, and then compute $\boldsymbol{\varepsilon}_{\text{PGD}}(\boldsymbol{\mu})$; another option is compute $\mathbf{PBU}^i\omega^i(\boldsymbol{\mu})$ by the external solver under the non-intrusive PGD scheme, resulting directly $\boldsymbol{\varepsilon}_{\text{PGD}}(\boldsymbol{\mu})$. Particularly in this example, the first option computes $\mathbf{B}$ only once, but due to the fact that the elemental strains are constant, extra post-processing is needed to compute the averaged strain thus this implementation is more intrusive. On the contrary, the second option needs to compute $\mathbf{B}$ for many times during the loops, so that we can use Abaqus to output strains that are already averaged, so the implementation could be more non-intrusive. In our case, we have adopted the second option for non-intrusiveness, and thus the result we have obtain is the computational vademecum of strains $\boldsymbol{\varepsilon}_{\text{PGD}}(\boldsymbol{\mu})$.

Remark 5 (On the identified parameters)
Regarding the resultant error norm $\|\boldsymbol{\varepsilon}^*_{\mathbb{P}} - \boldsymbol{\varepsilon}_{\mathbb{P}}(\alpha, \beta)\|$, its dependency on β is less sensitive in a small scale, this is because ρ^{e} raises the magnitude of $\mathbf{K}^{\text{e}}_1$ with about 10^3 in (53). The resulting parameters occur on the boundary $\alpha + \beta = 0$, this phenomenon indicates that the finite element model is not exactly reflecting the mechanics of real femur due to the simplifications we have made during the modelling process. On one hand, in the image-processing procedure, many empty spaces which should be eliminated during segmentation may have been included in the finite element mesh, therefore, the fact that they should have zero elastic modulus drives the identification result to the boundary. On the other hand, despite that the bone material might not be exact linear, the assumption that Young's modulus is linearly dependent on the intensity of CT image may have strongly increased the discrepancy of the FE model. Even so, the discrepancy between FE model and real sample does not disprove the effectiveness of the PGD vademecum.

6.4. Towards real-time simulation of mechanical response under variable loading locations

One of the most significant contributions of PGD computational vademecum is that we may take advantage of it for real-time simulations [19]. In clinical aspects, this kind of application is of great interest for patient-specific studies.

With the parameters identified in the former problem, we use the previously obtained material parameters $\boldsymbol{\mu}^* = (9.3589, -9.3589)$ for the model in this successive problem which takes only loading locations as the extra-coordinates, namely

$$\hat{\mathbf{U}}^n_{\mathrm{PGD}}(\mathbf{s}) = \sum_{i=1}^{n} \mathbf{U}^i \omega^i(\mathbf{s}). \tag{60}$$

A point load in $-z$ direction is applied on the femur head. The admissible domain of the loading locations is chosen as a finite surface on the femur head, i.e., $\mathbf{s} \in \bar{\Gamma}$. As already discussed in 3.2, to make the discretized parametric domain compatible with the finite element model, the surface is approximated to be composed by the exterior faces of the tetrahedral elements involved. Naturally, the DOFs of the parametric mesh just live on the sampled nodes.

Ideally, since $\dim \bar{\Gamma} = 2$, the parametric domain should be discretized by a 2D triangulation. To build the parametric mesh, we first use the graphical user interface (GUI) provided by Abaqus to extract the related nodes from the spatial mesh, and then use an in-house developed code (or even by hand) to reconstruct the triangular mesh, the procedure is illustrated in Figure 15. During the reconstruction process, proper renumbering is required to make the data compatible with the parent finite element model.

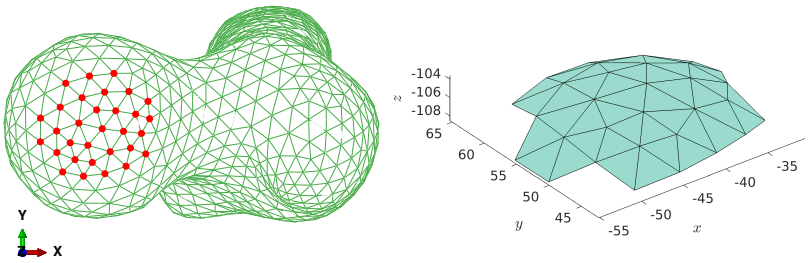


Figure 15. Extraction of the involved nodes (left) and reconstructed parametric mesh (right).

However, although practically the loaded nodes can only lie on the exterior surface of the femur head, mathematically it is allowed that we apply some "ghost" loads on nodes inside the solid. That is to say, the 2D parametric space can be extended to some tetrahedral elements which consist a subset of the whole 3D spatial domain, namely $\hat{\Omega} \subset \Omega$. With this dimensional extension, we increase the number of parameters from 2 to 3. The benefit of this extension is that the subset of tetrahedral elements can be directly extracted from the original finite element model. This extraction procedure, which is in a more non-intrusive fashion, can be performed within the handy Abaqus GUI as shown in Figure 16.

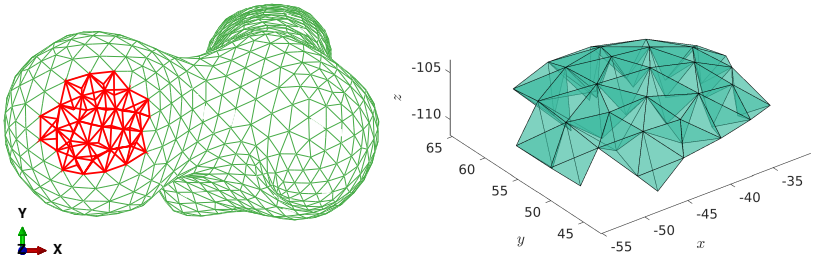


Figure 16. Parametric mesh (right) constructed by direct extraction of involved tetrahedral elements (left).

The second process does not require reconstruction of triangular mesh from node extraction, but would make the computation slightly more expensive. Considering the number of selected elements is usually small, the extra cost is acceptable. We computed the first 4 dominating modes with both meshes, the modal amplitude curves decrease at almost the same rate as plotted in Figure 17. The rate of convergence is nearly the same, which is not surprising because the parametric space

is discretized in the same order. The spatial modes represented with displacement in z direction computed with both parametric meshes are shown in Figure 18. The parametric modes computed with both meshes are shown in Figure 19. It is observed that the modes computed with the two parametric meshes are different both spatial and parametric, and the parametric modes computed with the 2D mesh are more spiny. Even so, the resultant displacements computed with each PGD vademecum is nearly the same. Compared to the standard FE result with a specified loading location, the error norm of the 2D parametric mesh result is 3.61% while that of the 3D parametric mesh result is 3.49%.

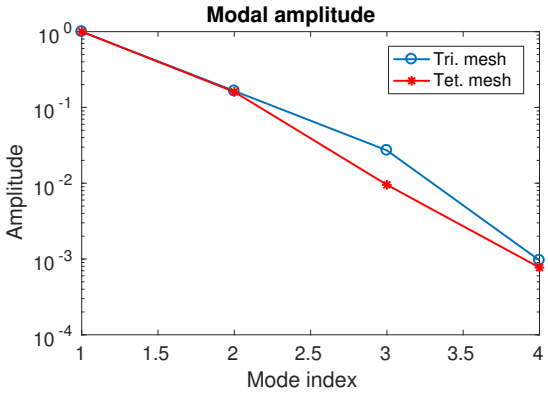


Figure 17. Normalized modal amplitude plot for triangular and tetrahedral parametric meshes.

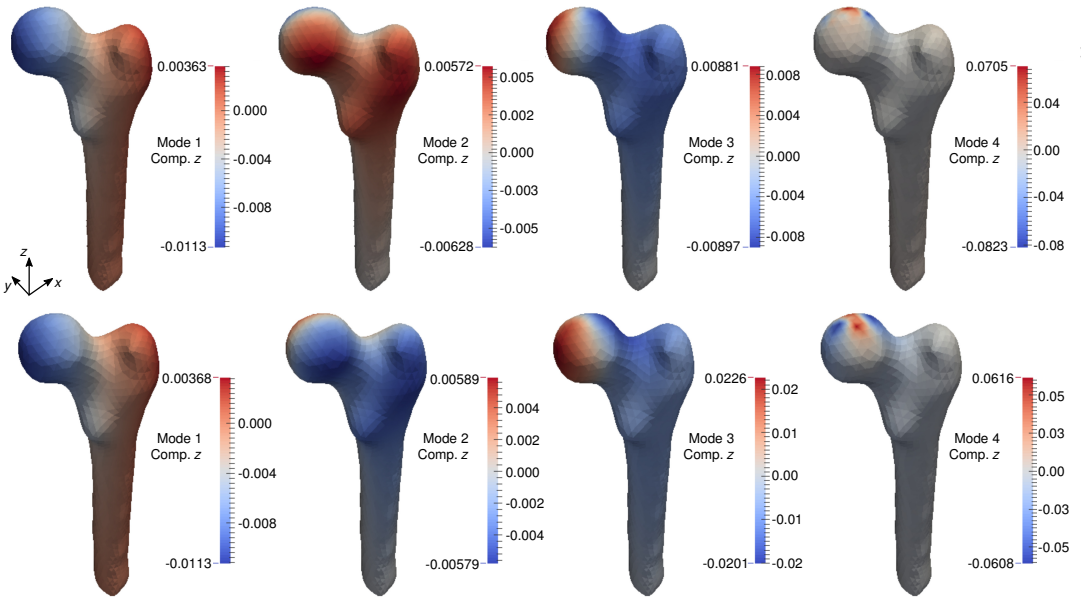


Figure 18. The z component of normalized spatial modes computed with triangular mesh (upper row) and tetrahedral mesh (lower row).

Having computed the computational vademecum $\hat{\mathbf{U}}_{\text{PGD}}(\mathbf{s})$, we are able to perform real-time simulation in the online computations on the numerical femur model. For instance, the strain field under different loading locations can be computed with (57). It is believed to have great potential clinic applications thanks to the fact that the online computation with the computational vademecum is extremely fast.

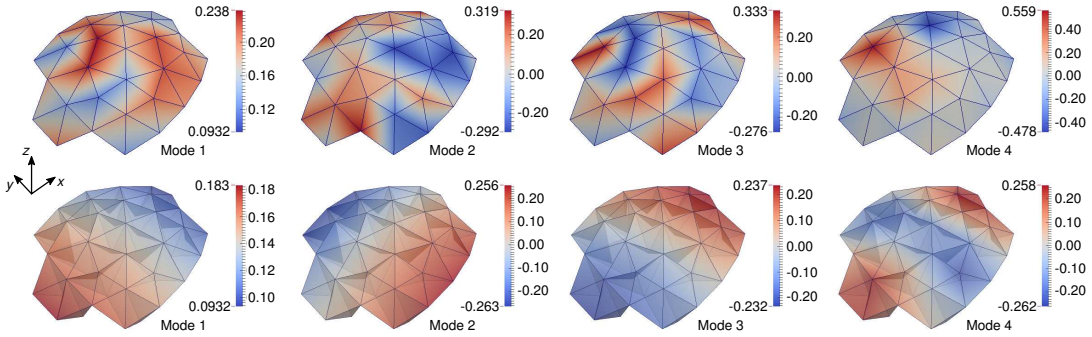


Figure 19. Parametric modes $\omega^j(\boldsymbol{\mu})$ computed with triangular mesh (upper row) and tetrahedral mesh (lower row).

7. CONCLUSIONS

In this work, we present a non-intrusive scheme for solving multi-dimensional PDEs with PGD. Taking linear elasticity problem as an example, the non-intrusive scheme is implemented with in-house developed Matlab codes calling Abaqus software as the external solver to solve the mechanical problems during the PGD offline phase. It is believed that for large-scaled multi-dimensional problems, the cost of developing and certifying PGD codes could be reduced by the implementation of the non-intrusive scheme. In the numerical examples, we implemented the non-intrusive PGD scheme taking material properties and loading locations as PGD extra-coordinates. Besides, similar works such as [18] also used geometrical parameters as PGD extra-coordinates.

Additionally, we propose a new strategy to collect together the parametric dimensions having mutual dependence. This extension also makes it possible to use PGD to solve parametric problems in constrained parameter spaces which are not separable in a Cartesian fashion.

Based on *in vitro* experiments, the linear model in bone mechanics we have chosen provided a comprehensive implementation of the techniques proposed in this paper. Such applications may have potentially great value for patient-specific simulation for clinic purposes.

ACKNOWLEDGEMENT

We acknowledge the support from the European Education, Audiovisual and Culture Executive Agency (EACEA) under the Erasmus Mundus Joint Doctorate *Simulation in Engineering and Entrepreneurship Development* (SEED), FPA 2013-0043.

Dr. M. Conti acknowledges European Research Council (ERC) through the ERC Starting Grant ISOBIO: Isogeometric Methods for Biomechanics (No. 259229).

REFERENCES

1. Ammar A. The proper generalized decomposition: A powerful tool for model reduction. *International Journal of Material Forming* 2010; **3**:89–102, doi:10.1007/s12289-009-0647-x.
2. Chinesta F, Ladevèze P ((eds.)). *Separated Representations and PGD-Based Model Reduction*, *CISM International Centre for Mechanical Sciences*, vol. 554. Springer Vienna: Vienna, 2014, doi:10.1007/978-3-7091-1794-1. URL http://link.springer.com/10.1007/978-3-7091-1794-1.
3. Chinesta F, Cueto E. *PGD-Based Modeling of Materials, Structures and Processes*. ESAFORM Bookseries on Material Forming, Springer International Publishing: Cham, 2014, doi:10.1007/978-3-319-06182-5. URL http://link.springer.com/10.1007/978-3-319-06182-5.
4. Chinesta F, Keunings R, Leygue A. *The Proper Generalized Decomposition for Advanced Numerical Simulations*. SpringerBriefs in Applied Sciences and Technology, Springer International Publishing: Cham, 2014, doi:10.1007/978-3-319-02865-1. URL http://link.springer.com/10.1007/978-3-319-02865-1.
5. Ammar A, Mokdad B, Chinesta F, Keunings R. A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids. *Journal of Non-Newtonian Fluid Mechanics* Dec 2006; **139**(3):153–176, doi:10.1016/j.jnnfm.2006.07.007. URL http://www.

sciencedirect.com/science/article/pii/S0377025706001662.
6. Ammar A, Mokdad B, Chinesta F, Keunings R. A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modelling of complex fluids. *Journal of Non-Newtonian Fluid Mechanics* Jul 2007; **144**(2-3):98–121, doi:10.1016/j.jnnfm.2007.03.009. URL http://www.sciencedirect.com/science/article/pii/S0377025707000821.
7. Rozza G. Reduced-basis methods for elliptic equations in sub-domains with a posteriori error bounds and adaptivity. *Applied Numerical Mathematics* Dec 2005; **55**(4):403–424, doi:10.1016/j.apnum.2004.11.004. URL http://linkinghub.elsevier.com/retrieve/pii/S0168927404002478.
8. Quarteroni A, Rozza G, Manzoni A. Certified reduced basis approximation for parametrized partial differential equations and applications. *Journal of Mathematics in Industry* Jun 2011; **1**(1):3, doi:10.1186/2190-5983-1-3. URL http://www.mathematicsinindustry.com/content/1/1/3.
9. Quarteroni A, Manzoni A, Negri F. *Reduced Basis Methods for Partial Differential Equations*, *UNITEXT*, vol. 92. Springer International Publishing: Cham, 2016, doi:10.1007/978-3-319-15431-2. URL http://link.springer.com/10.1007/978-3-319-15431-2.
10. Jolliffe IT. *Principal Component Analysis*. Springer Series in Statistics, Springer-Verlag: New York, 2002, doi:10.1007/b98835. URL http://link.springer.com/10.1007/b98835.
11. Nouy A. A priori model reduction through Proper Generalized Decomposition for solving time-dependent partial differential equations. *Computer Methods in Applied Mechanics and Engineering* 2010; **199**(23-24):1603–1626, doi:10.1016/j.cma.2010.01.009. URL http://dx.doi.org/10.1016/j.cma.2010.01.009.
12. Néron D, Ladevèze P. Proper Generalized Decomposition for Multiscale and Multiphysics Problems. *Archives of Computational Methods in Engineering* 2010; **17**(4):351–372, doi:10.1007/s11831-010-9053-2.
13. Chinesta F, Leygue A, Bordeu F, Aguado JV, Cueto E, Gonzalez D, Alfaro I, Ammar A, Huerta A. PGD-Based Computational Vademecum for Efficient Design, Optimization and Control. *Archives of Computational Methods in Engineering* 2013; **20**(1):31–59, doi:10.1007/s11831-013-9080-x. URL http://link.springer.com/article/10.1007/s11831-013-9080-x.
14. González D, Alfaro I, Quesada C, Cueto E, Chinesta F. Computational vademecums for the real-time simulation of haptic collision between nonlinear solids. *Computer Methods in Applied Mechanics and Engineering* Jan 2015; **283**:210–223, doi:10.1016/j.cma.2014.09.029. URL http://www.sciencedirect.com/science/article/pii/S0045782514003521.
15. Chinesta F, Ammar A, Leygue A, Keunings R. An overview of the proper generalized decomposition with applications in computational rheology. *Journal of Non-Newtonian Fluid Mechanics* Jun 2011; **166**(11):578–592, doi:10.1016/j.jnnfm.2010.12.012. URL http://www.sciencedirect.com/science/article/pii/S0377025711000061.
16. Chinesta F, Ladeveze P, Cueto E. A Short Review on Model Order Reduction Based on Proper Generalized Decomposition. *Archives of Computational Methods in Engineering* Oct 2011; **18**(4):395–404, doi:10.1007/s11831-011-9064-7. URL http://link.springer.com/10.1007/s11831-011-9064-7.
17. Cueto E, González D, Alfaro I. *Proper Generalized Decompositions*. SpringerBriefs in Applied Sciences and Technology, Springer International Publishing: Cham, 2016, doi:10.1007/978-3-319-29994-5. URL http://link.springer.com/10.1007/978-3-319-29994-5.
18. Courard A, Néron D, Ladevèze P, Ballere L. Integration of PGD-virtual charts into an engineering design process. *Computational Mechanics* Dec 2015; **57**(4):637–651, doi:10.1007/s00466-015-1246-y. URL http://link.springer.com/10.1007/s00466-015-1246-y.
19. Niroomandi S, González D, Alfaro I, Bordeu F, Leygue A, Cueto E, Chinesta F. Real-time simulation of biological soft tissues: A PGD approach. *International Journal for Numerical Methods in Biomedical Engineering* 2013; **29**(5):586–600, doi:10.1002/cnm.2544.
20. Niroomandi S, Alfaro I, González D, Cueto E, Chinesta F. Model order reduction in hyperelasticity: a proper generalized decomposition approach. *International Journal for Numerical Methods in Engineering* 2013; **96**(3):129–149, doi:10.1002/nme.4531. URL http://dx.doi.org/10.1002/nme.4531.
21. González D, Cueto E, Chinesta F. Computational Patient Avatars for Surgery Planning. *Annals of biomedical engineering* Jun 2015; **44**(1):35–45, doi:10.1007/s10439-015-1362-z. URL http://www.ncbi.nlm.nih.gov/pubmed/26101033.
22. Quesada C, González D, Alfaro I, Cueto E, Chinesta F. Computational vademecums for real-time simulation of surgical cutting in haptic environments. *International Journal for Numerical Methods in Engineering* 2016; **108**(10):1230–1247, doi:10.1002/nme.5252. URL http://doi.wiley.com/10.1002/nme.5252.
23. Chinesta F, Ammar A, Cueto E. Recent Advances and New Challenges in the Use of the Proper Generalized Decomposition for Solving Multidimensional Models. *Archives of Computational Methods in Engineering* 2010; **17**(4):327–350, doi:10.1007/s11831-010-9049-y.
24. Zlotnik S, Díez P, Modesto D, Huerta A. Proper generalized decomposition of a geometrically parametrized heat problem with geophysical applications. *International Journal for Numerical Methods in Engineering* Sep 2015; **103**(10):737–758, doi:10.1002/nme.4909. URL http://doi.wiley.com/10.1002/nme.4909.
25. Zlotnik S, Díez P, Gonzalez D, Cueto E, Huerta A. Effect of the separated approximation of input data in the accuracy of the resulting PGD solution. *Advanced Modeling and Simulation in Engineering Sciences* Nov 2015; **2**(1):28, doi:10.1186/s40323-015-0052-6. URL http://amses-journal.springeropen.com/articles/10.1186/s40323-015-0052-6.
26. González D, Aguado JV, Cueto E, Abisset-Chavanne E, Chinesta F. kPCA-Based Parametric Solutions Within the PGD Framework. *Archives of Computational Methods in Engineering* Mar 2016; doi:10.1007/s11831-016-9173-4. URL http://link.springer.com/10.1007/s11831-016-9173-4.
27. Al Akhras H, Elguedj T, Gravouil A, Rochette M. Towards an automatic isogeometric analysis suitable trivariate models generation—Application to geometric parametric analysis. *Computer Methods in Applied Mechanics and Engineering* 2017; **316**:623–645, doi:10.1016/j.cma.2016.09.030. URL http://www.sciencedirect.com/science/article/pii/S0045782516312221.

28. Mena A, Bel D, Alfaro I, González D, Cueto E, Chinesta F. Towards a pancreatic surgery simulator based on model order reduction. *Advanced Modeling and Simulation in Engineering Sciences* Nov 2015; **2**(1):31, doi: 10.1186/s40323-015-0049-1. URL http://www.amses-journal.com/content/2/1/31.
29. Siopack JS, Jergesen HE. Total hip arthroplasty. *The Western journal of medicine* Mar 1995; **162**(3):243–9. URL http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=1022709{&}tool=pmcentrez{&}rendertype=abstract.
30. Keaveny TM, Guo XE, Wachtel EF, McMahon TA, Hayes WC. Trabecular bone exhibits fully linear elastic behavior and yields at low strains. *Journal of Biomechanics* 1994; **27**(9):1127–1136, doi:10.1016/0021-9290(94)90053-1.
31. Yosibash Z, Padan R, Joskowicz L, Milgrom C. A CT-Based High-Order Finite Element Analysis of the Human Proximal Femur Compared to In-vitro Experiments. *Journal of Biomechanical Engineering* Nov 2006; **129**(3):297–309, doi:10.1115/1.2720906. URL http://dx.doi.org/10.1115/1.2720906.
32. Keller TS. Predicting the compressive mechanical behavior of bone. *Journal of biomechanics* Sep 1994; **27**(9):1159–68. URL http://www.ncbi.nlm.nih.gov/pubmed/7929465.
33. Taddei F, Pancanti A, Viceconti M. An improved method for the automatic mapping of computed tomography numbers onto finite element models. *Medical Engineering and Physics* Jan 2004; **26**(1):61–69, doi: 10.1016/S1350-4533(03)00138-3. URL http://www.sciencedirect.com/science/article/pii/S1350453303001383.
34. Nadal E, Chinesta F, Díez P, Fuenmayor F, Denia F. Real time parameter identification and solution reconstruction from experimental data using the Proper Generalized Decomposition. *Computer Methods in Applied Mechanics and Engineering* Nov 2015; **296**:113–128, doi:10.1016/j.cma.2015.07.020. URL http://www.sciencedirect.com/science/article/pii/S0045782515002327.